www.farelettronica.com
n. 340 – ottobre 2013

Come realizzare un circuito stampato in casa

Applicazioni USB con PIC18

Trasforma il notebook in un bromografo

Generatore di segnali

Con micro e display LCD

Controllo motore in PWM con dsPIC

Interfaccia grafica per Raspberry PI in Python

Sommario

340 - ottobre 2013

tutorial

progetti

rubriche

Editoriale

Imparare progettando

Guardando la copertina di questo numero la domanda può sorgere spontanea: perché un altro generatore di segnale? La risposta è semplice. Secondo la nostra filosofia, un progetto elettronico non deve essere solo utile nella sua realizzazione finale, ma deve anche in qualche modo arricchire la conoscenza di chi lo realizza. Non c'è modo migliore di imparare se non quello di mettere in pratica e toccare con mano ciò che fino ad un momento prima era scritto su carta. Il generatore di segnali pubblicato in questo numero non sono è uno strumento utile in laboratorio, ma è anche un progetto molto istruttivo per chi si accinge a realizzarlo. Viene realizzato in due versioni, una con PIC e una con Arduino, impiega moduli sintetizzatori Analog Devices ed un display LCD alfanumerico come interfaccia utente. Insomma, un concentrato di elettronica tutta da sperimentare. Cogliamo quindi l'occasione per rinnovare il nostro consueto invito: sperimentate, mettete in pratica e se qualche volta scoppia un condensatore… poco male, l'importante è avere imparato sempre qualcosa in più.

Buona lettura!

Maurizio Del Corso

DIRETTORE RESPONSABILE
Antonio Cirella

DIRETTORE TECNICO
Maurizio Del Corso

Segreteria di redazione
Giorgia Generali

Comitato scientifico
Simone Masoni (Microtest), Francesco Picchi (Microtest), Massimo Rovini (Università degli Studi di Pisa).

Grafica
Studio Punto & Virgola

Hanno collaborato a questo numero
Francesco Di Lorenzo
Giovanni Di Maria
Gaspare Santaera
Vincenzo Germano
Antonio Giannico
William Pasin
Maurizio Del Corso
Giuseppe Aquilani

Direzione e redazione
INWARE srl
Via Giotto, 7 - 20032 Cormano (MI)
Tel. 02.66504794
Fax 02.42101817
info@inwaredizioni.it
www.inwaredizioni.it
Redazione: fe@inwaredizioni.it

Pubblicità per l'Italia
Agostino Simone
Tel. +39 347 2230684
media@inwaredizioni.it

Europe and Americas
Elisabetta Rossi
Tel. +39 328 3245956
international@inwaredizioni.it

Asia
Cybermedia Communications Inc.
Tel. +886-(0)2-2691-2785
asia@inwaredizioni.it

Rest of the world
Inware srl
+39 02 66504794
info@inwaredizioni.it

Ufficio abbonamenti
INWARE srl
Via Giotto, 7 - 20032 Cormano (MI)
Per informazioni, sottoscrizione o rinnovo dell'abbonamento:
abbonamenti@inwaredizioni.it
Tel. 02.66504794
Fax. 02.42101817
L'ufficio abbonamenti è disponibile telefonicamente dal lunedì al venerdì dalle 14,30 alle 17,30.

• Membership MAKER:
11 numeri di Fare Elettronica in pdf € 29.99
• Membership MAKER FULL:
11 numeri di Fare Elettronica in pdf + Bonus Pack Maker € 39.99

Autorizzazione alla pubblicazione
Tribunale di Milano n.647
del 17/11/2003

Privacy
Nel caso la rivista sia pervenuta in abbonamento o in omaggio, si rende noto che i dati in nostro possesso sono impiegati nel pieno rispetto del D.Lgs. 196/2003. I dati trasmessi a mezzo cartoline o questionari presenti nella rivista, potranno venire utilizzati per indagini di mercato, proposte commerciali, o l'inoltro di altri prodotti editoriali a scopo di saggio. L'interessato potrà avvalersi dei diritti previsti dalla succitata legge. In conformità a quanto disposto dal Codice di deontologia relativo al Trattamento di dati personali art. 2, comma 2, si comunica che presso la nostra sede di Cormano Via Giotto 7, esiste una banca dati di uso redazionale. Gli interessati potranno esercitare i diritti previsti dal D.Lgs. 196/2003 contattando il Responsabile del Trattamento Inware Edizioni Srl (info@inwaredizioni.it).

Collaborare con FARE ELETTRONICA
Le richieste di collaborazione vanno indirizzate all'attenzione di Maurizio Del Corso (m.delcorso@inwaredizioni.it) e accompagnate, se possibile, da una breve descrizione delle vostre competenze tecniche e/o editoriali, oltre che da un elenco degli argomenti e/o progetti che desiderate proporre.

Elenco inserzionisti

News

Toshiba Electronics presenta i nuovi MOSFET di potenza miniaturizzati a supergiunzione da 600 V

Toshiba Electronics Europe ha annunciato che la nuova generazione di MOSFET di potenza a supergiunzione (SJ) in tecnologia DTMOS-IV sarà disponibile per la prima volta in contenitori DFN piccoli e sottili. I nuovi MOSFET DTMOS-IV da 600 V con sistema di incapsulamento DFN saranno particolarmente adatti per realizzare alimentatori ad elevata velocità di commutazione, ballast per lampade a scarica e altre applicazioni che richiedono alternative dall'ingombro ridotto ai tradizionali formati D2PAK e DPAK. Con correnti nominali nell'intervallo da 9,7 A fino a oltre i 30 A, i nuovi dispositivi della famiglia di MOSFET da 600 V TKxV60W possiedono una bassissima resistenza di conduzione (RDS(ON)) nel range da 0,38 Ω a soli 0,098 Ω. L'eccellente cifra di merito RDS(ON)*Qg garantisce una commutazione altamente efficiente, mentre la piccola capacità di uscita (Coss) permette un funzionamento ottimizzato per i carichi leggeri. Entrambi i dispositivi comprendono anche un ulteriore contatto sensore per il collegamento diretto al circuito di pilotaggio. Il processo DTMOS-IV di Toshiba permette di realizzare dei MOSFET che presentano per RDS(ON) un coefficiente di temperatura migliore dei dispositivi alternativi. Ciò permette di ottenere una maggiore efficienza anche con elevate temperature di funzionamento. Come con gli altri dispositivi Toshiba ad architettura DTMOS-IV, i nuovi MOSFET DFN possiedono una capacità (Cgd) di gate-drain ottimizzata che offre un migliore controllo di commutazione dv/dt. Il supporto a valori inferiori di dv/dt contribuisce inoltre a ridurre la tendenza all'effetto di oscillazione (ringing) in circuiti di commutazione ad alta velocità. Con dimensioni di 8 mm x 8 mm, il contenitore DFN presenta un ingombro superficiale del 20% più piccolo rispetto a un contenitore D2PAK. Lo spessore di soli 0,85 mm è quasi tre volte più sottile rispetto al tradizionale DPAK e oltre cinque volte più sottile rispetto al D2PAK. Toshiba utilizza il processo a trincea profonda nel MOSFET DTMOS-IV a supergiunzione di quarta generazione con tecnologia monoepitassiale. Ciò permette di ottenere un passo di trincea più stretto rispetto al processo (multiepitassiale) di terza generazione, con una resistenza di conduzione migliore del 30% per una specifica superficie del die.

www.toshiba-components.com

News

Micrel lancia due LED driver ad alta corrente ed alta efficienza per il flash delle fotocamere con calibrazione precisa della corrente

Micrel, Inc. ha annunciato il lancio di due nuovi LED driver ad alta corrente per il flash delle fotocamere. I modelli MIC2870 e MIC2871 sono in grado di azionare uno o due LED per il flash della fotocamera, fino a 1,5 A, con massima efficienza e precisa calibrazione della corrente. Questi dispositivi sono ideali per applicazioni portatili che funzionano a batteria. Il MIC2870 opera in un intervallo di tensione da oltre 2,7 a 5,5 V ed è in grado di azionare uno o due LED fino a 1,5 A. Il MIC2871 opera ad un intervallo di tensione da oltre 2,7 a 5,5 V ed è in grado di azionare un unico LED fino a 1,2 A. La corrente del LED per il flash stroboscopico e la luce del video può essere configurata in 16 livelli diversi attraverso l'I2C (MIC2870) o l'interfaccia unifilare (MIC2871).
Questi driver LED per fotocamera funzionano ad una frequenza di commutazione di 2 MHz e forniscono una soluzione ad alta potenza ed estremamente compatta con un'efficienza fino al 95%. Entrambi i dispositivi sono dotati di varie caratteristiche di sicurezza, come il rilevamento della carica bassa della batteria, la sconnessione del carico in uscita, il timer di sicurezza, la protezione da sovratensione, la protezione da cortocircuito del LED, l'inibitore del flash e protezione termica.

www.micrel.com

I disaccoppiatori a schiera a bassa induttanza serie LICA di AVX sono ideali per le applicazioni mil/aero

AVX Corporation ha ampliato le applicazioni dei disaccoppiatori a schiera a bassa induttanza serie LICA. Ampiamente utilizzati nelle applicazioni commerciali e industriali di tipo 'five nines (99.999%) uptime' come i CPU di classe mainframe, i moduli multi-chip di massime performance e i sistemi di comunicazioni, i disaccoppiatori a schiera a bassa induttanza seria LICA, testati secondo gli standard MIL-PRF-123, forniscono inoltre soluzioni avanzate di di-

News

saccoppiamento ad alta affidabilità e con induttanza estremamente bassa per molteplici applicazioni militari e aerospaziali, inclusi: FPGA ad alto conteggio di pin, ASIC, CPU e altri circuiti integrati ad alta potenza e alte performance con basse tensioni di funzionamento. Caratterizzato da un valore di ESL di < 30pH, che è attualmente il rating più basso di ESL nel mercato del settore mil/aero, la serie LICA di AVX include varie configurazioni progettuali a bassa induttanza.
Utilizzando fino a quattro sezioni di condensatori separati in un'unica capsula ceramica, le schiere della serie LICA sono dotate di elettrodi in platino a bassa resistenza in configurazione di basso rapporto altezza-larghezza, doppi pickup di elettrodo e percorsi della corrente perpendicolari. Tale serie costituisce inoltre l'unico prodotto del mercato mil/aero che presenta terminazioni a piastrina C4 (C4 solder ball), che permettono un'induttanza di interconnessione minimale e che sono il perfetto completamento della tecnologia di packaging della piastrina senza adduttori.

www.avx.com

WEB facile: Mouser velocizza la progettazione con il tool TI WEBENCH Designer

Mouser Electronics, Inc. offre ora uno strumento di progettazione software online Texas Instruments che crea progetti personalizzati in pochi secondi. WEBENCH Designer consente ai progettisti di generare, ottimizzare e simulare progetti conformi alle loro specifiche esclusive.

Questo strumento di progettazione TI coniuga le diverse esigenze a livello di progetto, sistema e supply chain prima che il progetto stesso passi alla fase di produzione. WEBENCH è ideale per una gran varietà di progetti, inclusi alimentazione, illuminazione, filtri, clock e rilevamento. Mouser Electronics supporta il progetto completo grazie alla vastità dei prodotti TI e di altri fornitori leader disponibili a magazzino. WEBENCH restituisce una distinta materiali completa.

www.mouser.com/new/Texas-Instruments/TI-Webench-Designer

Controller SuperSpeed USB3 Microchip disponibile da Mouser

Mouser Electronics, Inc. tiene ora a magazzino il nuovo hub controller SuperSpeed USB3 certificato di Microchip, il primo al mondo con memoria programmabile on-chip e hub a 4 porte

News

USB3 certificato con 3 lane USB2 supplementari. La famiglia Microchip USB553XB-5000 costituita da quattro modelli è la soluzione più flessibile del settore in quanto include una versione ibrida a 7 porte con hub USB3 certificato a 4 porti e tre lane USB2 supplementari. Gli altri tre modelli della famiglia presentano rispettivamente hub USB3 rispettivamente a due, tre e quattro porte per un ampio percorso di migrazione a disposizione di progettisti di PC, periferiche, piattaforme di computer, soluzioni di storage e dispositivi consumer, decoder, docking station e monitor.
I progettisti di piattaforme di elaborazione e periferiche stanno rapidamente adottando lo standard USB3 per la sua maggiore larghezza di banda di 5 Gbps e porte di carica a maggiore potenza. Questa famiglia di prodotti riduce i costi di sistema e velocizza lo sviluppo integrando memoria programmabile che consente agli OEM di personalizzare rapidamente le soluzioni e garantisce compatibilità con il sistema operativo senza i costi aggiuntivi, la complessità e lo spazio su scheda richiesto dalla memoria off-chip. Inoltre, il controller ibrido offre la flessibilità necessaria per creare applicazioni con porte USB3 e USB2, quali dispositivi mobili e docking station per PC, monitor e altre periferiche. La disponibilità di componenti a due, tre e quattro porte USB3 consente ai progettisti di migrare facilmente al migliore UCH3 per ciascun progetto all'interno della propria piattaforma.

www.mouser.com/applications/usb30/

www.mouser.com/new/microchip-usb5532b-controller/

Anritsu Company introduce nuove opzioni per PIM Master

Anritsu introduce tre opzioni di frequenza per il suo pluripremiato PIM Master MW82119A, l'unico analizzatore di PIM portabile e ad alta potenza, alimentato a batteria. Con queste nuove opzioni, gli ingegneri e i tecnici che lavorano sul campo potranno usufruire di un apparecchio portatile in grado di eseguire misurazioni PIM anche nelle bande LTE 800 MHz e 2.600 MHz, e nella banda UMTS 2.100 MHz.

L'intermodulazione passiva (PIM) è un problema significativo per l'LTE, soprattutto quando vengono attivate contemporaneamente le nuove frequenze di diversi operatori in siti adiacenti o nello stesso sito, o nei vecchi siti dove sono già in funzione le tecnologie 2G e 3G.
Le nuove opzioni sono state specificatamente sviluppate per tutte le bande assegnate alla rete LTE in Europa, per aiutare gli operatori a minimizzare le interferenze PIM che possono causare un basso throughput di dati, e quindi una bassa qualità del servizio percepito dagli utenti.
Assieme al funzionamento a batteria, le dimensioni compatte e la possibilità di eseguire test PIM a 40 W, le nuove opzioni fanno del PIM Master MW82119A lo strumento ideale per l'implementazione, l'installazione e la manutenzione delle reti LTE che utilizzano soluzioni RRH sulle torri oltre che per le installazioni tradizionali e i sistemi DAS.

www.anritsu.com

Eventi

FIERA DELL'ELETTRONICA DI LIVORNO

Espositori provenienti da tutta Italia proporranno prodotti che spazieranno dai consumabili per pc (Dvd e Cd vergini, cartucce per stampanti e kit di ricarica), periferiche, accessori per pc, lettori multimediali, apparecchiature radio, microcamere, impianti fotovoltaici, articoli per spionaggio, lampade a led, telefonia anche con ricambi, ferramenta, batterie ricaricabili, giocattoli e gadget elettronici e tanto altro ancora.
*Rispetto ai megastore di distribuzione elettronica verranno offerti prodotti di grande qualità a prezzi incredibili, è proprio il caso di dire "**Hi teck low cost**". Sarà inoltre presente la sezione ARI (associazione radioamatori italiani) di Livorno con uno stand. Un appuntamento imperdibile per gli appassionati di tecnologia ... e per chi vuole risparmiare!*

Dove: Livorno • Quando: 23-24 novembre 2013 • Orari: dalle 9.00 alle 18.00 •
Organizzazione: Prometeo • info: www.prometeo.tv

EXPOELETTRONICA FORLÌ

Alla Fiera di Forlì appuntamento con uno dei più importanti eventi nazionali "consumer" di settore: quello con la Grande Fiera dell'Elettronica che da dicembre 2008 ha cambiato il nome in Expo Elettronica. Le cifre sono quelle delle grandi occasioni: circa 200 espositori, 20.000 mq di esposizione e circa 29.000 visitatori attesi alla manifestazione.Expo Elettronica ha un pubblico vasto ed eterogeneo: appassionati del "fai da te", elettro-riparatori, "smanettoni", radioamatori, "cacciatori" di buone occasioni o pezzi rari; questo perché propone un panorama merceologico e un calendario di eventi collaterali veramente ricchissimo.

Dove: Forlì (FC) • Quando: 7-8 dicembre 2013 • Orari: dalle 9.00 alle 18.00 •
Organizzazione: Blunautilus • info: www.expoelettronica.it

FIERA MERCATO DELL'ELETTRONICA

Qualità, basso costo dei prodotti venduti ed INGRESSO GRATUITO per i possessori della Fidelity Card, contraddistinguono le fiere organizzate da Printmix. Troverete tutto quello di cui avete bisogno per quanto riguarda l'elettronica: pc, notebook, microcomponenti, cartucce, cd/dvd o accessori. Il tutto a prezzi convenienti, senza trascurare l'ottima qualità che gli espositori presenti sono sempre in grado di offrirvi.

Dove: Mariano C.se (CO) • Quando: 21-22 dicembre 2013 • Orari: dalle 9.00 alle 18.00 •
Organizzazione: Printmix • info: www.fieramercatodellelettronica.it

vuoi far parte del CLUB? Sei MAKER, SMART o GENIUS?

Entra anche tu nel mondo dell'elettronica con Inware Edizioni!

CLUB MEMBERSHIP

Club Light
accesso e/o download agli articoli (oltre 300) del Club di Firmware

Club Full
accesso e/o download agli articoli (oltre 700) del Club di Fare Elettronica

GENIUS MEMBERSHIP

- 11 nuovi numeri di Fare Elettronica in PDF
- 11 nuovi numeri della rivista digitale Firmware (i numeri andranno in coda all'eventuale abbonamento esistente)
- tutto l'archivio delle riviste Firmware pubblicate mensilmente dal febbraio 2010

Bonus Pack Genius (solo per la card Genius Full)

- accesso e/o download agli articoli (oltre 300) del Club di Firmware
- accesso e/o download agli articoli (oltre 700) del Club di Fare Elettronica
- due ebook a scelta* (Smartcard, CPLD, Display LCD, PIC, Operazionali, alimentatori, linguaggio C, Basic per PIC, Elettronica Analogica)
- due raccolte delle annate di Fare Elettronica in PDF (dal 2003 al 2012) e Firmware (dal 2006 al 2012).

SMART MEMBERSHIP

- 11 nuovi numeri della rivista digitale Firmware (i numeri andranno in coda all'eventuale abbonamento esistente)
- tutto l'archivio delle riviste Firmware pubblicate mensilmente dal febbraio 2010

Bonus Pack Smart (solo per la card Smart Full)

- accesso e/o download agli articoli (oltre 300) del Club di Firmware
- un ebook a scelta* (Smartcard, CPLD, Display LCD, PIC, Operazionali, alimentatori, linguaggio C, Basic per PIC, Elettronica Analogica)
- una raccolta delle annate di Firmware in PDF (dal 2006 al 2012)

MAKER MEMBERSHIP

- 11 nuovi numeri di Fare Elettronica in PDF (i numeri andranno in coda all'eventuale abbonamento esistente)

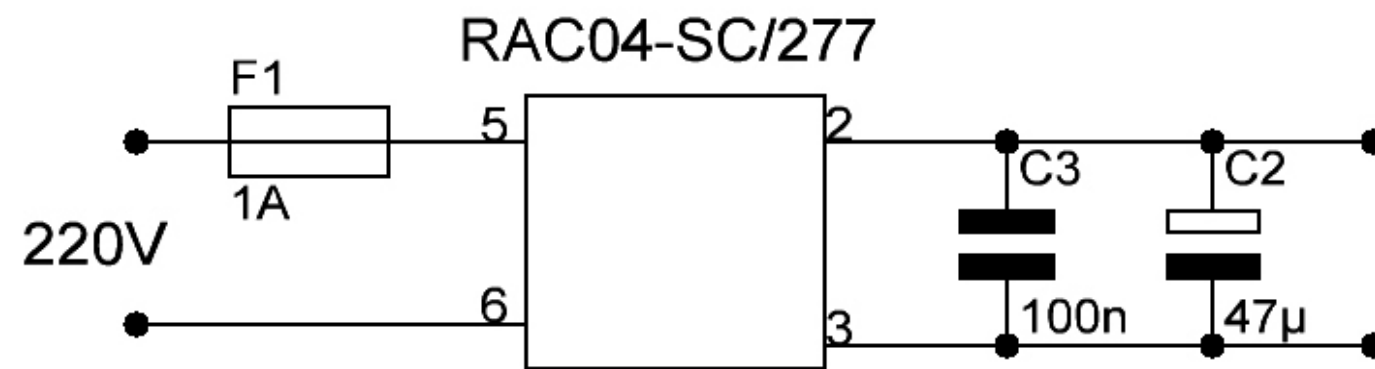
Bonus Pack Maker (solo per la card Maker Full)

- accesso e/o download agli articoli (oltre 700) del Club di Fare Elettronica
- un ebook a scelta* (Smartcard, CPLD, Display LCD, PIC, Operazionali, alimentatori, linguaggio C, Basic per PIC, Elettronica Analogica)
- una raccolta delle annate di Fare Elettronica in PDF a scelta (dal 2003 al 2012)

SCEGLI LA TUA MEMBERSHIP E FARAI PARTE ANCHE TU DEL CLUB PIÙ ESCLUSIVO DELL'ELETTRONICA!

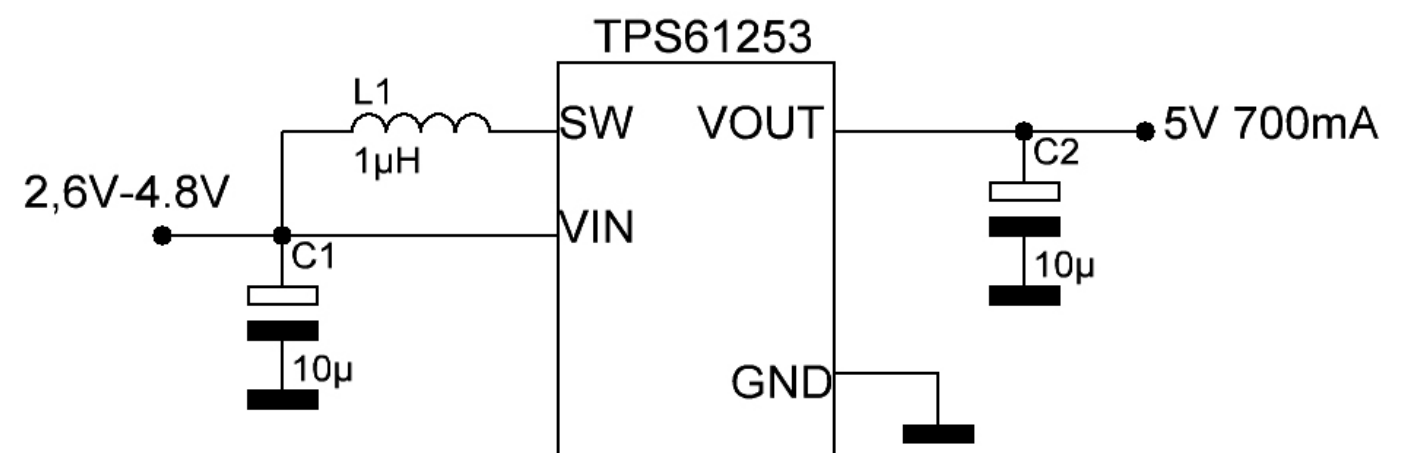
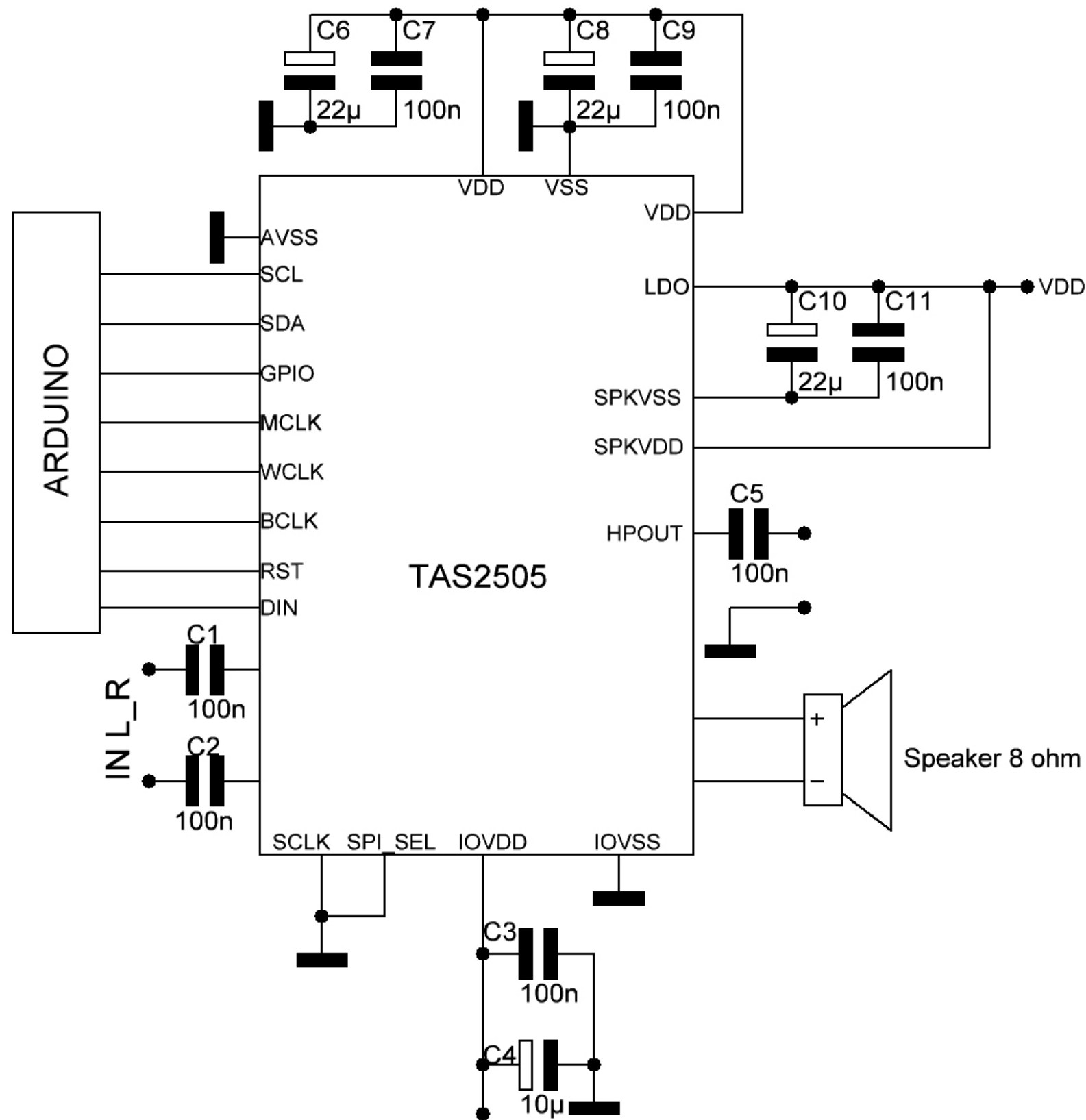
I prezzi delle card si intendono IVA inclusa

*la scelta viene effettuata al ricevimento della card

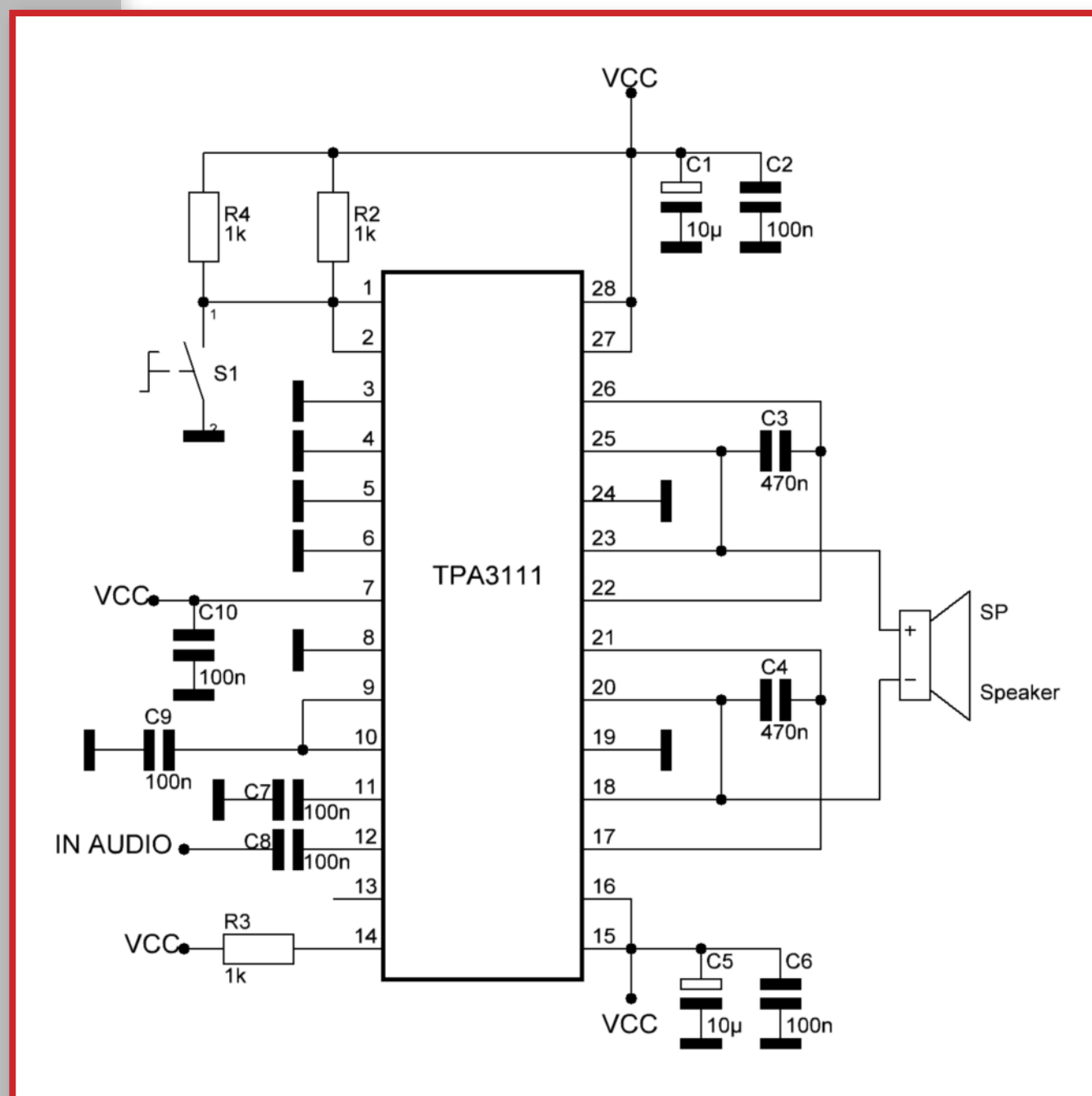
Idee di progetto

CONVERTITORE AC/DC

Lo schema riportato in figura è un convertitore AC/DC il quale permette di ottenere tensioni continue di diversi valore ingombrando poco spazio. Tale circuito ,vista la sua semplicità di connessione, viene molto impiegato in applicazioni relative alle telecomunicazioni.

CARICABATTERIA PER CELLULARE

Lo schema in figura è un caricabatteria per cellulari, realizzato con un TPS61253, il quale è un semplicissimo convertitore DC/DC che permette di ottenere una corrente massimale di 700mA e con tensione di 5V, fornendo in uscita una potenza massima di 3,5W.

Idee di progetto

AMPLIFICATORE CLASSE D

In figura è riportato un semplice circuito realizzato con un amplificatore di classe D, che permette la gestione dei vari parametri mediante l'utilizzo di un microcontrollore come ARDUINO. L'integrato utilizzato per questa applicazione è un TAS2505, il quale deve essere alimentato con una tensione massima di 5V

Idee di progetto

AMPLIFICATORE DI CLASSE D CONNESSIO A BTL

L'amplificatore in figura è in grado di erogare una potenza massima di 10W con un carico di 8ohm. Tale circuito trova largo uso nei dispositivi audio portatili. Vista la presenza di un regolatore di tensione interno tale circuito può essere alimentato con una tensione che va dagli 8V ai 26V.

di GIOVANNI DI MARIA

GUI IN PYTHON PER RASPBERRY PI

In questo articolo il lettore imparerà a creare una applicazione Gui per Raspberry Pi, utilizzando il linguaggio di programmazione Python

Negli articoli precedenti abbiamo imparato a realizzare piccole applicazioni pratiche, con la possibilità di comandare ed acquisire dati ed informazioni dalle porte esterne GPIO del Raspberry Pi. Il livello di interfacciamento con l'utente è diversificato, a seconda del linguaggio di programmazione utilizzato, infatti:
utilizzando il linguaggio C standard, l'interfaccia grafica è completamente assente ed il colloquio con l'utente si limita a semplici visualizzazioni di stringhe e dati sullo schermo, senza particolari effetti grafici;
utilizzando il linguaggio Harbour, il colloquio con l'utente è più piacevole e suggestivo. Vi è una interfaccia semigrafica che consente l'utilizzo di colori, menù di scelta e inserimento dati dentro campi specifici, posizionati in qualunque punto dello schermo.

Figura 1: Interfaccia utente creata con il linguaggio C

Utilizzando opportuni linguaggi di programmazione, ma ancor di più adatte e specifiche librerie, è possibile programmare le proprie applicazioni in modo che esse utilizzano l'interfaccia GUI come canale di comunicazione con l'utente. Tale modalità infatti oggi è uno standard e consente una maggiore funzionalità ed usabilità senza paragoni. Con essa l'interazione è garantita attraverso l'uso del mouse e sono consentite tutte le operazioni grafiche e di visualizzazione True Color.

VARI MODI PER PROGRAMMARE LE GUI CON RASPBERRY PI

La creazione di un interfaccia GUI non è proprio una operazione semplice, come quella della scrittura di un programma a linea di comando. Occorre infatti utilizzare alcune librerie esterne o moduli da importare, a seconda di quale linguaggio si utilizza.
Programmando il Raspberry Pi, e quindi in Linux, si hanno molte soluzioni. L'articolo approfondirà solo una metodologia, che poi è quella più semplice e meno dispendiosa in termini di risorse e tempi di sviluppo.

TKINTER

Il modulo Tkinter è una interfaccia standard del linguaggio Python per la programmazione delle GUI. La sua programmazione somiglia lontanamente a quella

Figura 2: Una elegante Interfaccia utente realizzata con il linguaggio Harbour

delle classi QT. Esso è infatti un insieme di oggetti GUI per questo linguaggio. Abbiamo scelto tale modulo per alcuni motivi ben precisi:
Il modulo Tkinter è già presente e preinstallato in tutte le distribuzioni di Linux con Python (e quindi anche in quella del Raspberry Pi);
La sua semplicità è estrema e con poche righe di programmazione si riesce a realizzare un buon prodotto finale;
E' portabile, pertanto il codice può essere utilizzato con altri sistemi operativi. Naturalmente le chiamate dirette ai files proprietari del Raspberry Pi (vedi i files che indirizzano le porte di comunicazione) non saranno interpretate correttamente.
Per maggiori informazioni sull'utilizzo delle classi Tkinter invitiamo il lettore a documentarsi in rete.

Per eseguire il programma, l'utente deve avviare obbligatoriamente l'ambiente grafico, digitando il comando **startx** al prompt dei comandi. Le finestre GUI sono infatti una prerogativa di X e non dell'interfaccia testuale.

LA MIA PRIMA FINESTRA GUI

E' sorprendente notare come con tre sole linee di codice si possa realizzare una finestra GUI. Per iniziare a prendere confidenza con il sistema si scrivano le seguenti linee di codice, utilizzando l'editor preferito:

```
from Tkinter import *    #01
rad = Tk()               #02
rad.mainloop()           #03
```

Si salvi quindi il file con il nome di "prova01.py". L'estensione "py" indica appunto che il sorgente è stato scritto per il linguaggio Python. Il numero progressivo ad inizio di ciascuna riga è solo per ottenere un riferimento nella spiegazione, ma non si deve inserire in fase di codifica definitiva.
Attenzione: in alcuni sistemi occorre sostituire il nome del modulo "Tkinter" con "tkinter", in minuscolo.
Segue una breve spiegazione del listato.
La prima riga di programma ha lo scopo di importare il modulo Tkinter. Quindi viene creata una istanza alla classe Tk, con nome di "rad" (o qualunque altro nome desiderato). Tale istanza rappresenta la finestra principale della nostra applicazione. L'ultima funzione serve per "bloccare" il programma ed attende gli eventi generati (spostamento, ridimensionamento, uscita, pressione pulsanti, tastiera, ecc).
Dopo aver memorizzato, basta invocare (con qualsiasi sistema) il Python specificando il nome del programma sorgente realizzato in precedenza.
Lo scopo si può raggiungere scegliendo una delle seguenti soluzioni:

- Premendo il tasto "K" (lo Start di Windows), scegliendo "Run" e digitando il comando: "python prova01.py". E' consigliabile specificare il secondo parametro completo di percorso delle directories;

Figura 3: Editing del primo sorgente di Python

- Avviando Python e richiamando il programma creato da questa sede;
- In ambiente "X" richiamando la finestra terminale e digitando il comando: "python prova01.py". Questa soluzione è ottima in quanto permette di controllare eventuali errori del programma sulla console;
- Creando e richiamando un script con il comando di avvio;
- Creando sul desktop o in un'altra cartella un link al programma;

E molti altri...

Ovviamente è necessario posizionarsi nella directory che contiene il sorgente, altrimenti bisogna specificare per intero il path completo.

PILOTIAMO UN DIODO LED

Bene, ora che abbiamo familiarizzato con la creazione di una finestra GUI siamo, per così dire, pronti a realizzare una applicazione più completa. Questo prototipo consiste in un semplice programma, formato da due pulsanti che, rispettivamente, comandano l'accensione o lo spegnimento di un diodo Led, collegato ad una delle uscite GPIO del Raspberry Pi.

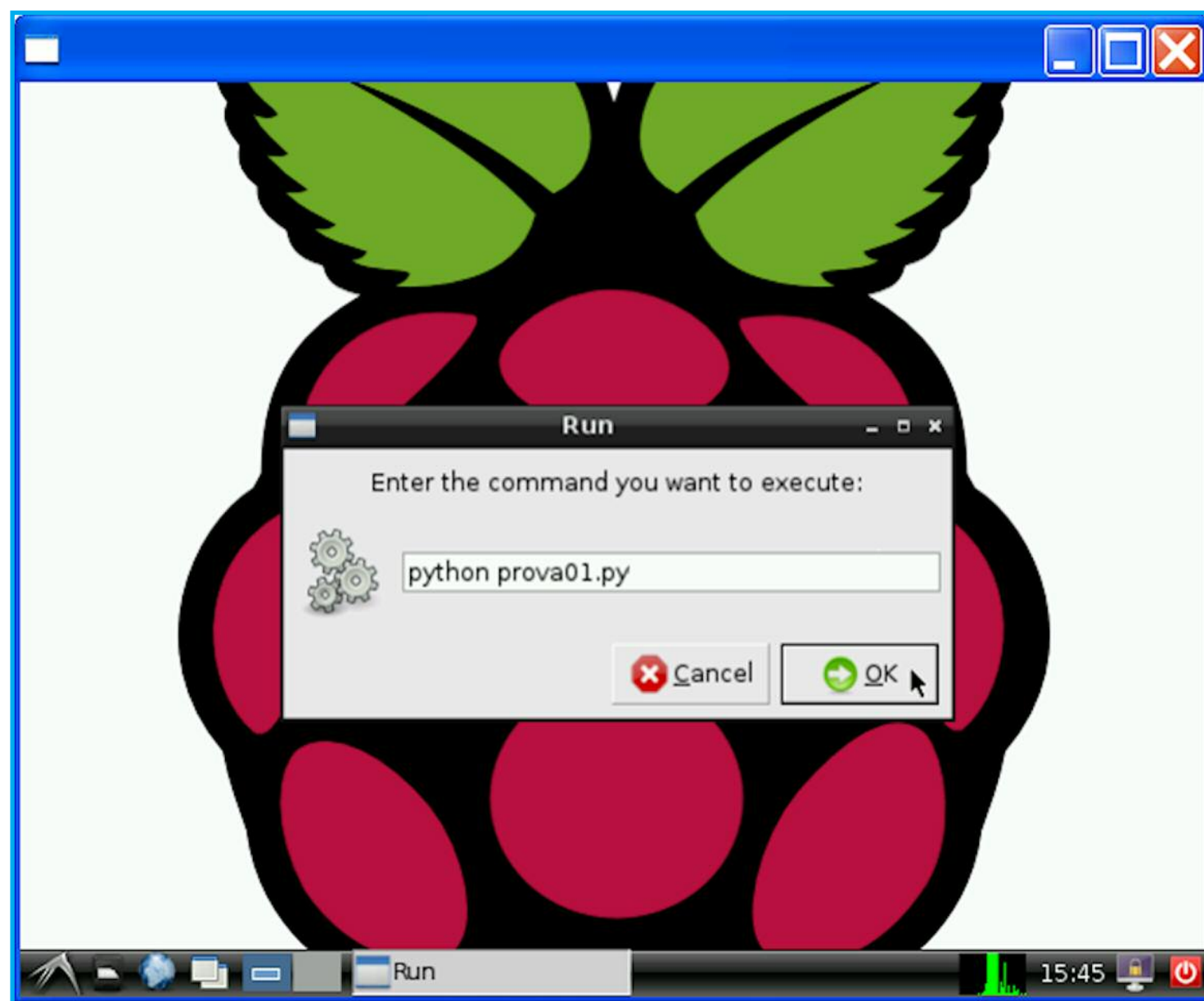



Figura 4: Uno dei tanti metodi per eseguire il proprio programma in Python

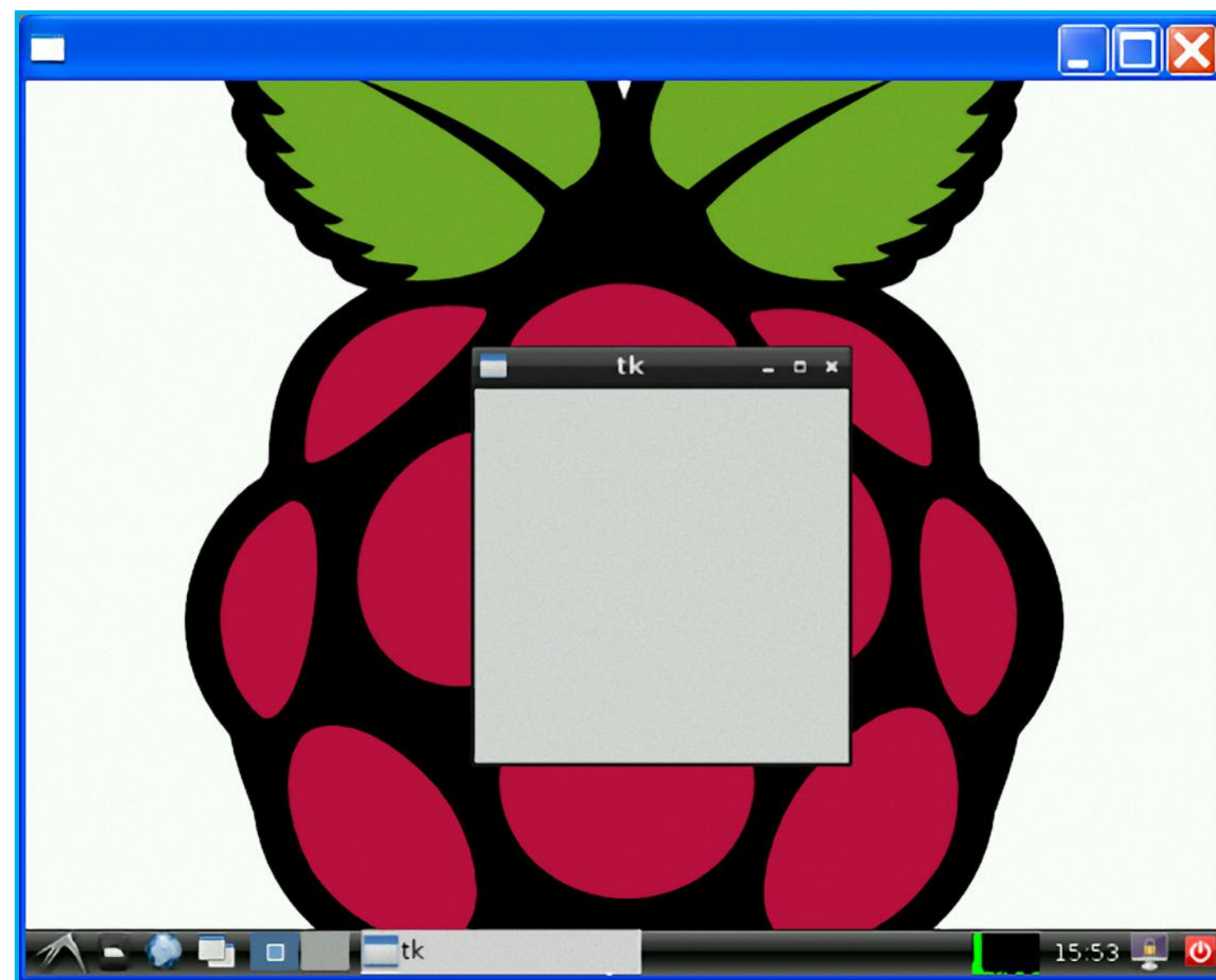



Figura 5: La mia prima finestra GUI

Schema elettrico

Lo schema elettrico è semplicissimo e richiama i primi esempi affrontati in questo tutorial, ai quali rimandiamo il lettore. Il diodo Led è collegato alla porta di uscita GPIO4 (pin 7 del Raspberry Pi) tramite la solita resistenza di limitazione. Ovviamente tale porta è configurata come uscita, lo vedremo tra poco nella codifica del software.

Il software

Il programma non è molto lungo, ma tante saranno le soddisfazioni nel vederlo girare perfettamente come applicazione GUI nella finestra di X. Al solito, non occorre digitare il numero progressivo a fine riga, esso serve solo come riferimento della spiega-

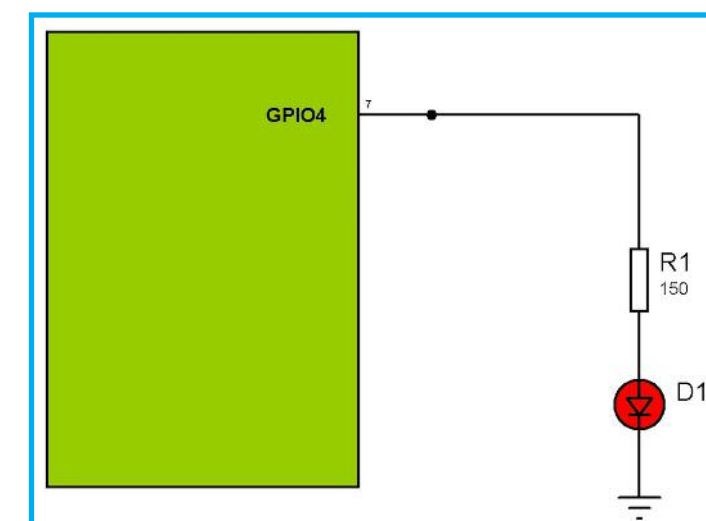



Figura 6: Un diodo Led collegato sulla porta GPIO4

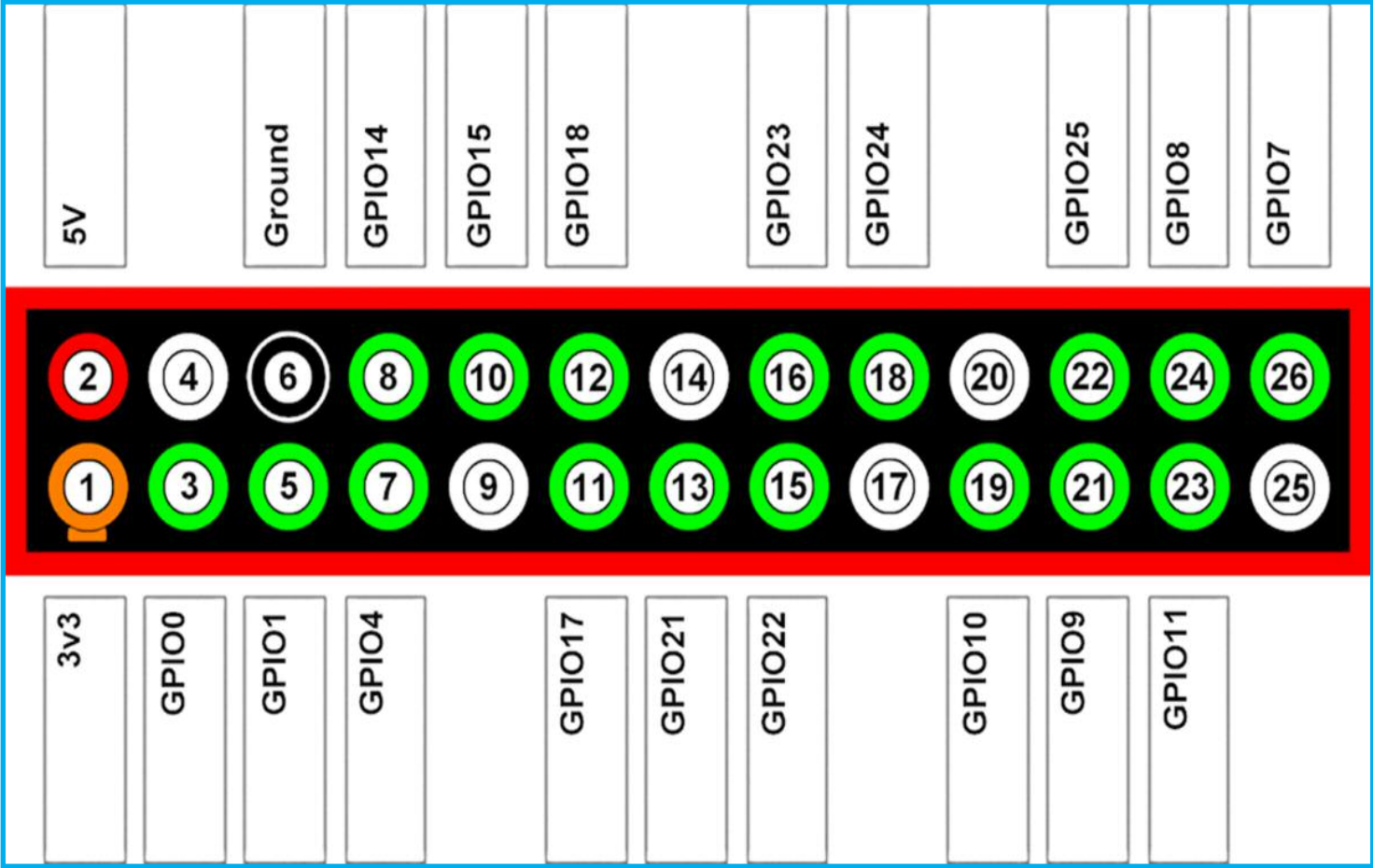



Figura 7: La porta GPIO del Raspberry Pi

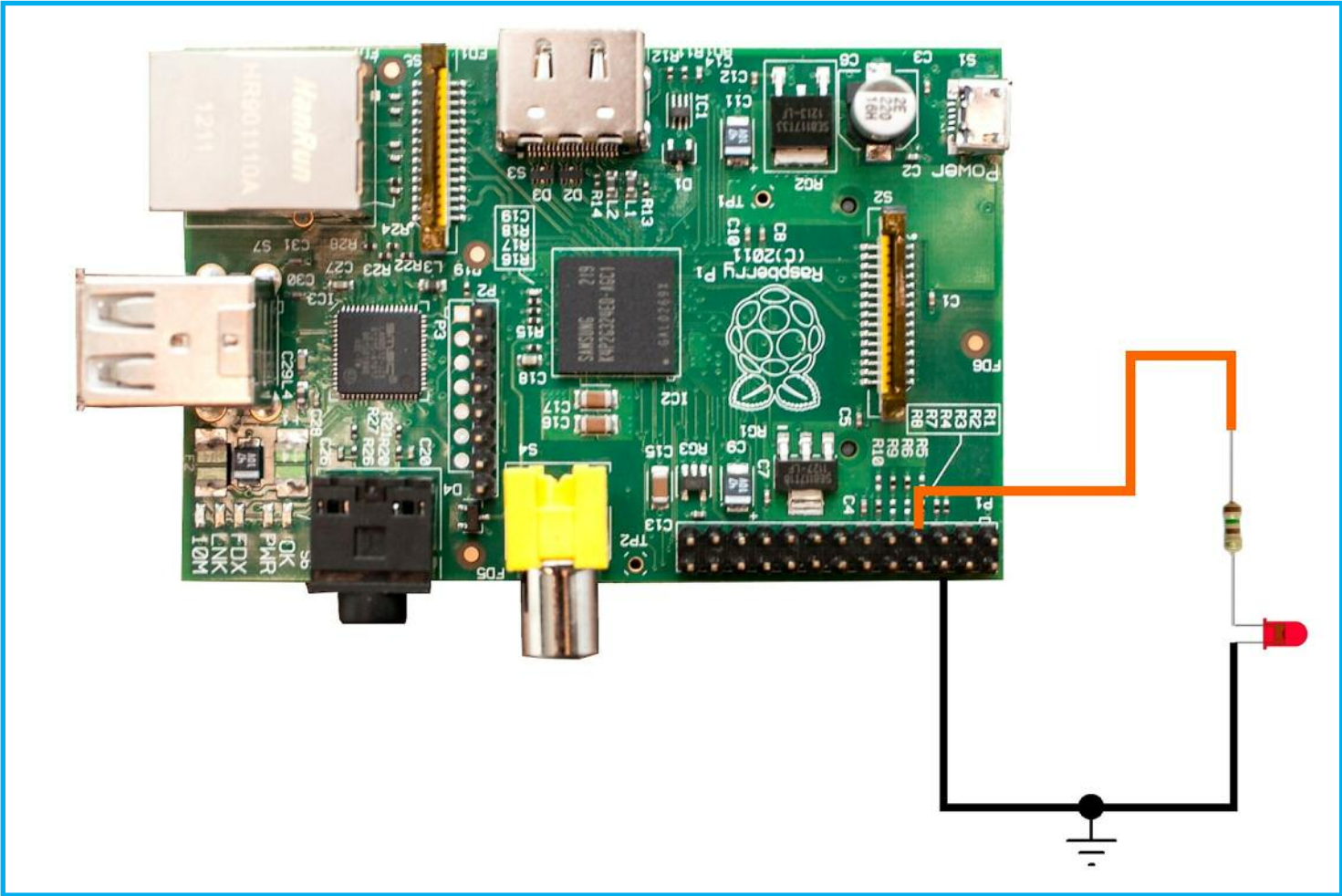



Figura 8: Connessione di un diodo led alla porta GPIO4 del Raspberry Pi

LISTATO 1

Download

```
from Tkinter import *                                          #01
                                                               #02
def LedOn():                                                   #03
  f=open("/sys/class/gpio/gpio4/value","w")                    #04
  f.write("1")                                                 #05
  f.close()                                                    #06
                                                               #07
def LedOff():                                                  #08
  f=open("/sys/class/gpio/gpio4/value","w")                    #09
  f.write("0")                                                 #10
  f.close()                                                    #11
                                                               #12
radice=Tk()                                                    #13
radice.title("Pulsanti con Raspberry Pi")                      #14
radice.geometry("280x100")                                     #15
                                                               #16
pulsante1=Button(radice,command=LedOn)                         #17
pulsante1["text"]="Accendi Led"                                #18
pulsante1["background"]="#BBFFBB"                              #19
pulsante1.place(x=20,y=20)                                     #20
                                                               #21
pulsante2=Button(radice,command=LedOff)                        #22
pulsante2["text"]="Spegni Led"                                 #23
pulsante2["background"]="#FFBBBB"                              #24
pulsante2.place(x=150,y=20)                                    #25
                                                               #26
f=open("/sys/class/gpio/export","w")                           #27
f.write("4")                                                   #28
f.close()                                                      #29
f=open("/sys/class/gpio/gpio4/direction","w")                  #30
f.write("out")                                                 #31
f.close()                                                      #32
                                                               #33
radice.mainloop()                                              #34
f=open("/sys/class/gpio/unexport","w")                         #35
f.write("4")                                                   #36
f.close()                                                      #37
```

zione. Nello scrivere il sorgente in linguaggio Python, occorre prestare estrema attenzione alla indentazione del codice.
Non esistendo infatti contrassegni particolari che racchiudano i blocchi logici (come parentesi graffe o begin-end), la struttura del programma è determinata esclusivamente dai rientri di riga delle varie istruzioni.
Commentiamo adesso il codice.

- La riga 1 del programma ha lo scopo di importare il modulo Tkinter.
- Le righe da 3 a 6 definiscono la funzione "LedOn" e scrivono il valore "1" nel file "/sys/class/gpio/gpio4/value", provocando l'accensione del diodo led collegato sulla porta GPIO4.
- Le righe da 8 a 11 definiscono la funzione "LedOff" e scrivono il valore "0" nel file "/sys/class/gpio/gpio4/value", provocando lo spegnimento del diodo led collegato sulla porta GPIO4.
- Le righe da 13 a 15 costruiscono la finestra principale, chiamata "radice". Si crea infatti una istanza alla classe Tk, con nome di "radice" (o qualunque altro nome desiderato). Tale istanza rap-

Figura 10: L'applicazione GUI in funzione

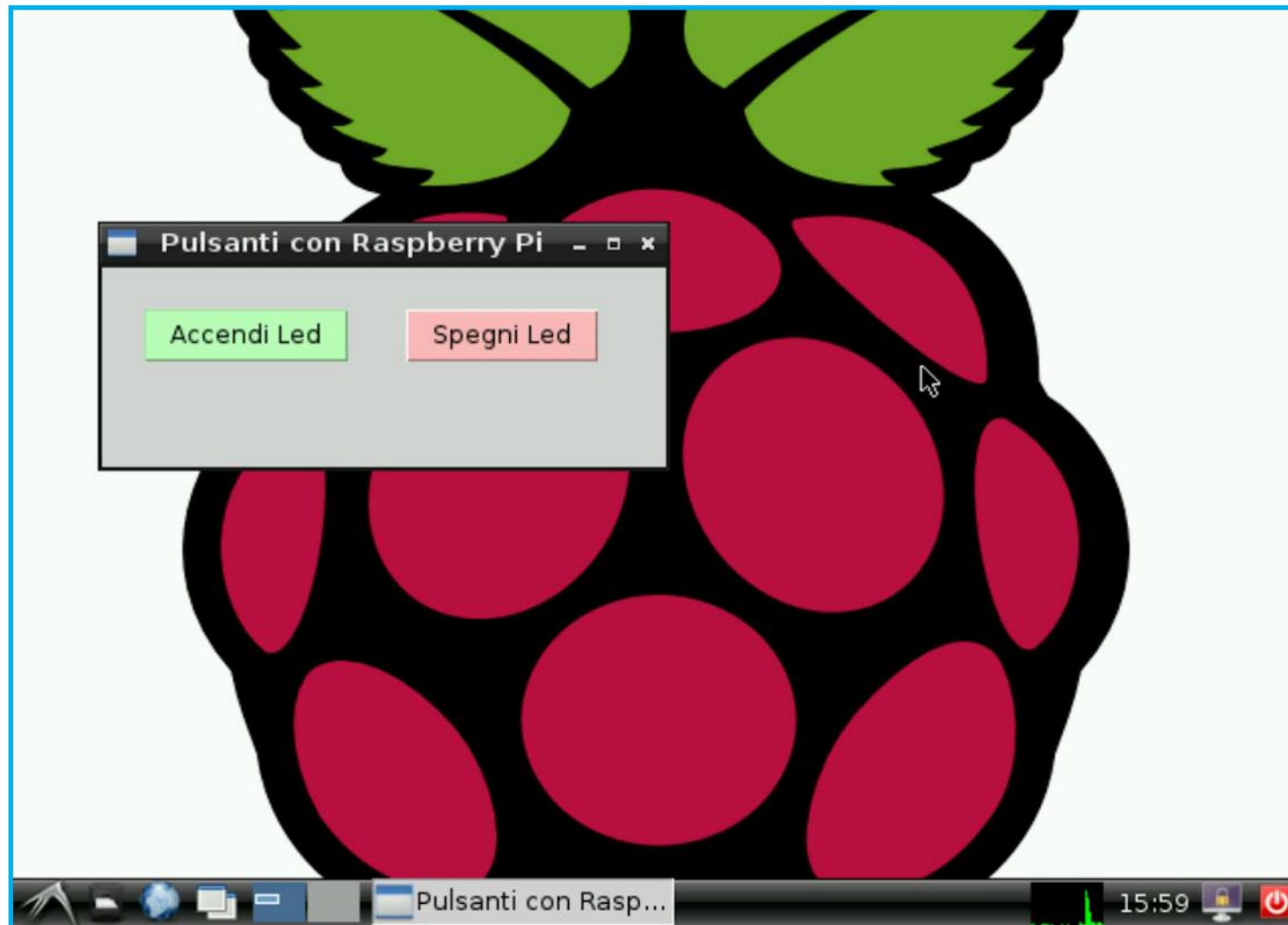



Figura 9: L'applicazione GUI che comanda il diodo Led

LISTATO 2

Download

```
from Tkinter import *                                    #01
                                                         #02
def LedOn():                                             #03
    label.configure(image=photo1)                        #04
    f=open("/sys/class/gpio/gpio4/value","w")            #05
    f.write("1")                                         #06
    f.close()                                            #07
                                                         #08
def LedOff():                                            #09
    label.configure(image=photo2)                        #10
    f=open("/sys/class/gpio/gpio4/value","w")            #11
    f.write("0")                                         #12
    f.close()                                            #13
                                                         #14
radice=Tk()                                              #15
radice.title("Un Led con Raspberry Pi")                  #16
radice.geometry("320x240")                               #17
                                                         #18
pulsante1=Button(radice,command=LedOn)                   #19
pulsante1["text"]="Accendi Led"                          #20
pulsante1["background"]="#BBFFBB"                       #21
pulsante1.place(x=20,y=20)                               #22
                                                         #23
pulsante2=Button(radice,command=LedOff)                  #24
pulsante2["text"]="Spegni Led"                           #25
pulsante2["background"]="#FFBBBB"                       #26
pulsante2.place(x=200,y=20)                              #27
                                                         #28
photo1 = PhotoImage(file="led-on.gif")                   #29
photo2 = PhotoImage(file="led-off.gif")                  #30
                                                         #31
label = Label(image=photo2)                              #32
label.place(x=120,y=80)                                  #33
                                                         #34
f=open("/sys/class/gpio/export","w")                     #35
f.write("4")                                             #36
f.close()                                                #37
f=open("/sys/class/gpio/gpio4/direction","w")            #38
f.write("out")                                           #39
f.close()                                                #40
                                                         #41
radice.mainloop()                                        #42
                                                         #43
f=open("/sys/class/gpio/unexport","w")                   #44
f.write("4")                                             #45
f.close()                                                #46
```

presenta dunque la finestra principale della nostra applicazione. Vengono anche definiti il titolo della finestra e le dimensioni.

- Le righe da 17 a 20 costruiscono il primo pulsante, che ha la funzione di accendere il diodo led.
- Le righe da 22 a 25 costruiscono il secondo pulsante, che ha la funzione di spegnere il diodo led. Entrambi i pulsanti, se premuti, richiamano rispettivamente le funzioni "LedOn" e "LedOff".
- Le righe da 27 a 32 sono eseguite una volta sola, nell'ambito dell'intero programma, ed hanno il compito di definire la porta GPIO4. Le prime tre istruzioni (27, 28, 29) creano un riferimento alla porta GPIO4 mentre le successive tre istruzioni (30, 31,32) ne definiscono la direzione di uscita ("out").
- La riga 34 pone il programma in stato di attesa, per ogni eventuale segnale esterno generato (spostamento, ridimensionamento, uscita, pressione pulsanti, tastiera, ecc).
- Le righe da 35 a 37 sono eseguite all'uscita del programma, ossia quando l'utente chiude l'applicazione con la "X" di chiusura, e hanno lo scopo di "cancellare" la porta GPIO4 in modo che il programma (es il sistema operativo) non la consideri più.

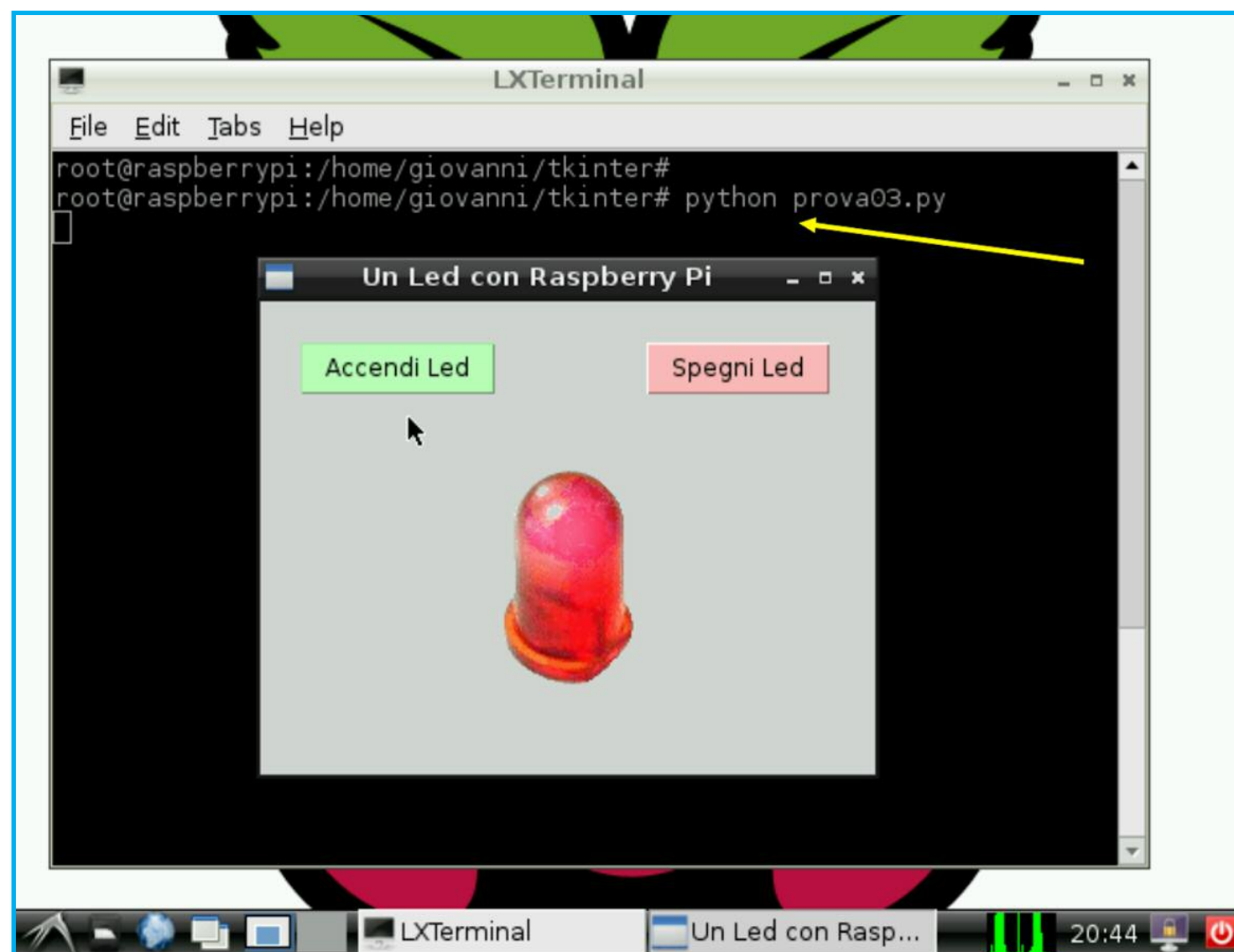



Figura 11: L'applicazione GUI con immagini interattive

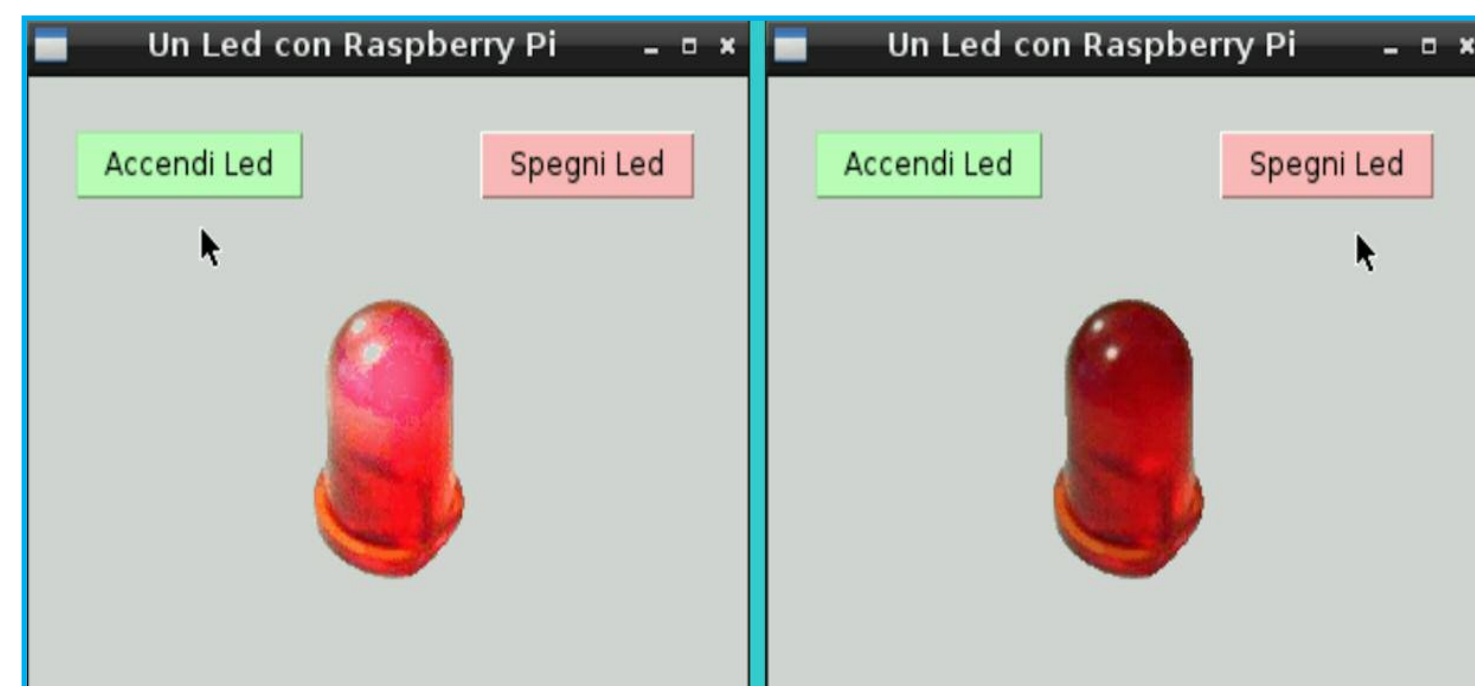



Figura 12: L'applicazione GUI nei due diversi stati logici

PILOTAGGIO DI UN LED CON IMMAGINI INTERATTIVE

Perfezioniamo l'esempio precedente aggiungendo sulla finestra del programma l'immagine di un diodo led. Essa segue le stesse condizioni del diodo led "reale", alternando lo stato di acceso e di spento anche sulla GUI del monitor o della televisione.

Ovviamente occorre apportare qualche modifica al programma, in modo da inglobare anche gli oggetti di tipo immagine, assegnati ad una label. Le immagini sono memorizzate nella stessa directory del programma, con tipologia GIF e sfondo trasparente. Il nome della prima immagine è "led-on.gif" mentre quello della seconda è "led-off.gif".

Utilizzando degli elementi grafici che cambiano o modificano il proprio stato, in dipendenza di eventi interni o esterni, è indubbiamente più professionale ed aumenta maggiormente il livello interattivo con l'applicazione finale.

Commentiamo adesso il codice, focalizzando solo le differenze con il precedente, per quanto riguarda l'implementazione delle immagini grafiche:

- La riga 4 "carica" la foto1 come immagine, corrispondente al diodo led acceso;
- La riga 10 "carica" la foto2 come immagine, corrispondente al diodo led spento;
- Le righe 29 e 30 caricano le immagini utilizzando la classe PhotoImage;
- Le righe 32 e 33 creano e posizionano una Label, contenente inizialmente la foto2 (led spento).

CONCLUSIONI

Questo articolo costituisce un importantissimo tassello nella programmazione del Raspberry Pi. Con esso si è infatti potuto apprendere la tecnica di base per produrre software con una interfaccia GUI, oggigiorno utilizzata più di qualunque altro tipo di tecnica di colloqui uomo-macchina.

La libreria RPi.GPIO per Python

Per coloro che non vogliono programmare direttamente le porte a basso livello, esiste un modulo per Python da scaricare ed importare, che permette l'accesso alla GPIO in modo molto semplice e veloce, ottenendo anche la drastica riduzione del codice sorgente. Essa mette a disposizione una classe con la quale è possibile appunto controllare la porta GPIO.
La libreria si chiama Raspbian package for Python (python rpi gpio). Segue un semplice esempio di utilizzo.

```
import RPi.GPIO as GPIO

# to use Raspberry Pi board pin numbers
GPIO.setmode(GPIO.BOARD)

# set up GPIO output channel
GPIO.setup(12, GPIO.OUT)

# set RPi board pin 12 high
GPIO.output(12, GPIO.HIGH)

# input from RPi board pin 11
input_value = GPIO.input(11)
```

Per avviare una applicazione GUI occorre ricordare di avviare l'ambiente X di Linux, digitando il comando **startx** e, soprattutto, di effettuare l'accesso al sistema come **root**.
Realizzare una interfaccia grafica GUI, con relativamente poche linee di codice, è un'impresa estremamente soddisfacente e mette davanti una serie di idee e progetti molto più performanti, rispetto alle equivalenti possibilità offerte dal modo testo a console.
Questa possibilità può infatti prevedere l'utilizzo di grafica, immagini, foto e filmati che arricchiscono indubbiamente un'applicazione utente.
Per maggiori dettagli sulle porte di input e di output si consiglia di leggere i precedenti articoli di Fare Elettronica, sullo studio e la sperimentazione sul Raspberry Pi.
La filosofia di questa tipologia di tutorial è quella che afferma che basta saper illuminare un diodo led collegato ad una porta di un controllore, MCU o altro, per produrre qualunque tipologia di applicazioni.
Buon lavoro.

di GIOVANNI DI MARIA

CANDELABRO VOTIVO A MICROCONTROLLORE

Ecco un progetto dalle mille soluzioni di applicazione: un microcontrollore, che prevede un interruttore temporizzato, con cui si possono attivare dei carichi indipendenti per un certo periodo di tempo. Una classica attuazione di questo tipo si riscontra con i candelabri votivi, utilizzati nei luoghi di culto

L'aspetto didattico interessante del progetto è che le lampade sono indipendenti. Ciò vuol dire che quando si aziona una lampada per alcuni minuti, si può avviare in qualsiasi momento un'altra lampada, indipendentemente dallo stato logico delle altre.
In quasi tutte le Chiese e i luoghi di culto sono presenti uno o più candelabri votivi. In questo articolo vedremo come realizzare la parte logica e decisionale del dispositivo, utilizzando un microcontrollore. Ormai l'elettronica è entrata anche in Chiesa a grandi passi. Molti dispositivi sacri e liturgici si sono evoluti in altrettanti congegni meccanici ed elettronici. Basti pensare al "Rosario elettronico portatile" che aiuta il fedele a recitare le preghiere, con tanto di altoparlante che suggerisce le varie litanie e lampadine per scandire le varie stazioni. Esistono anche quadretti sacri elettronici, con musiche e luci automatiche, candele elettroniche computerizzate, Vie Crucis elettroniche, Vangeli elettronici automatizzati, ecc. Anche le musiche liturgiche, molto spesso, vengono comandate dal sacerdote che, con un semplice pul-

Figura 1: Un classico candelabro votivo elettronico

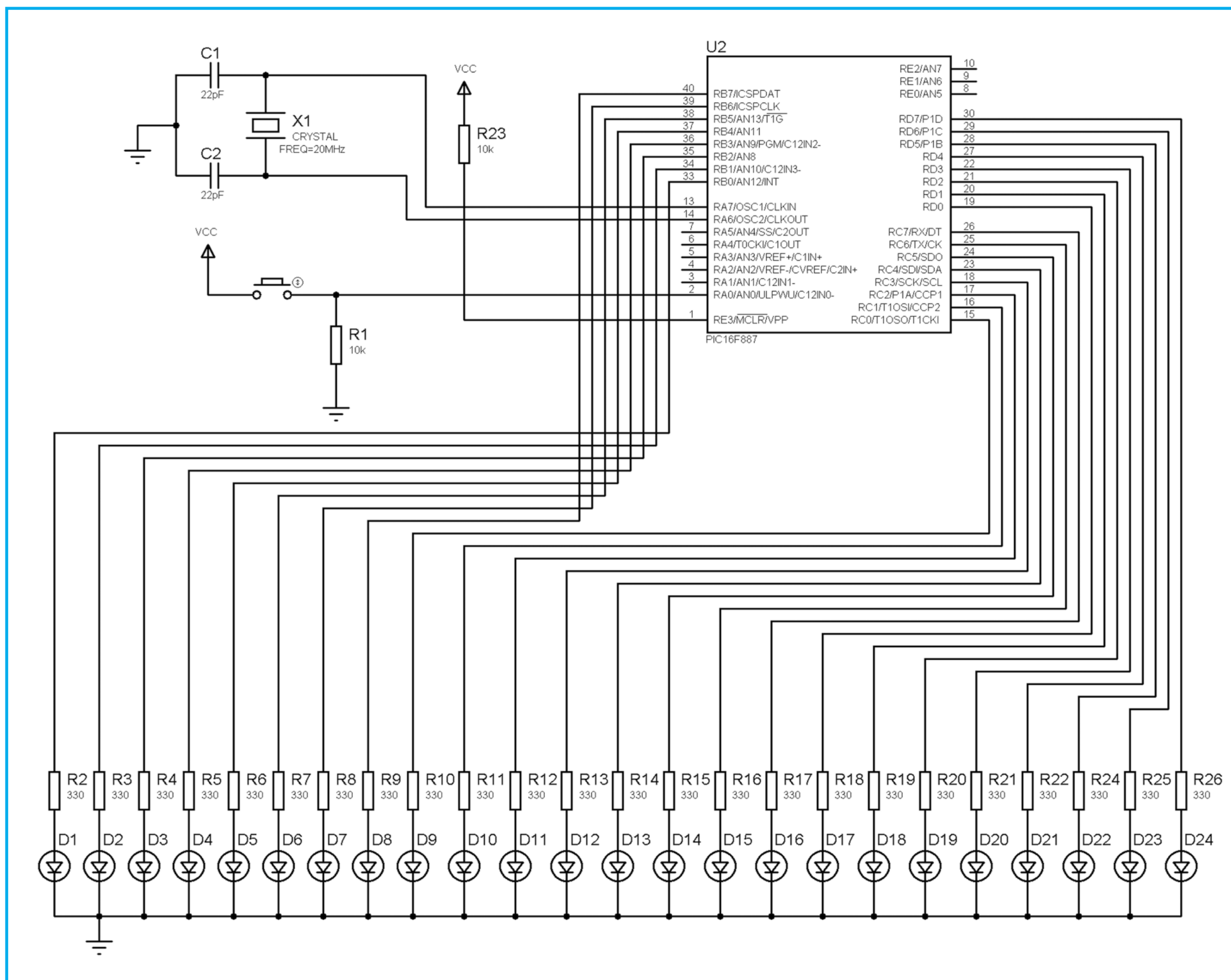



Figura 2: Schema elettrico

sante poggiato sull'altare, avvia o interrompe la riproduzione delle stesse.
In questo articolo ci dedichiamo alla progettazione di un candelabro votivo prediligendo, come sempre, l'aspetto didattico e istruttivo del circuito e della programmazione del microcontrollore.
Si tratta sostanzialmente di un classico "interruttore a gettone" che comanda, per un certo lasso di tempo, un determinato carico.
Ma il suo utilizzo può essere esteso a tantissime altre applicazioni pratiche, come vedremo nel proseguo dell'articolo.

IL CANDELABRO VOTIVO

Il candelabro votivo, conosciuto anche come candeliere votivo o candeliere elettronico automatico, ha un ruolo importante. Il suo funzionamento si basa su candele ecologiche elettroniche (lampadine o led). Con l'inserimento di una moneta, una candela si accende e resta in tale stato per un determinato periodo di tempo.
Si tratta quindi di un candeliere elettronico, con accensione e spegnimento automatico ed indipendente, che scongiura del tutto il pericolo di un incendio dovuto alla combustione accidentale e incontrollata delle candele.

LA DIFFICOLTÀ

Dal punto di vista della programmazione, questa applicazione risulta leggermente difficoltosa. L'architettura del PIC, non disponendo di un sistema operativo e, inoltre, non prevedendo il multitasking, rende complessa la gestione indipendente (e a volte simultanea) della temporizzazione dei carichi. Ognuno deve essere attivato in qualsiasi istante e la durata della sua attivazione deve essere la stessa per tutte le uscite, indipendentemente dallo stato logico delle altre. Ciò nonostante, ricorrendo a qualche accorgimento algoritmico, il problema viene "brillantemente" risolto.

ANALISI DI FUNZIONAMENTO

Nella sua struttura essenziale, il prototipo deve funzionare nel modo seguente:

- Inizialmente tutte le lampade sono spente;

- Se il fedele inserisce una moneta (alias: preme il pulsante) si accende, per un tempo determinato, la prima lampada libera e disponibile (ossia fino a quel momento spenta);
- Nel frattempo, un'ulteriore pressione del pulsante accende un'altra lampada disponibile, e così via;
- Ogni lampada resta accesa per un certo numero di secondi, è il suo funzionamento è indipendente dalle altre, come se il sistema sia dotato di tanti microcontrollori per ogni carico.

SCHEMA ELETTRICO

Lo schema elettrico, a differenza del firmware implementato all'interno della CPU, è molto semplice. Si tratta di un microcontrollore PIC 16F887 con un ingresso logico localizzato sulla porta RA0. Esso simula l'immissione della moneta e può essere naturalmente sostituito con un reale, ma costoso, inseritore a moneta. Il resto delle porte costituisce l'uscita logica del circuito, tutte rivolte al pilotaggio dei carichi, come led, lampade o relè. Nel caso di questi ultimi occorre adottare un modulo di potenza poiché il micro, da solo, non è in grado di reggere il "peso" di ben 24 carichi. Il microcontrollore è coadiuvato, come al solito, del quarzo da 20 Mhz e dai due condensatori da 22pF.

IL LISTATO

Il listato, anche se relativamente corto, è un po' complesso. In pratica, per ognuno dei 24 carichi, è stato previsto un vettore numerico (array) che contiene il numero dei centesimi di secondo disponibili affinché esso abbia esaurito la sua funzione. In pratica, non appena il pulsante viene premuto (ovvero è stata inserita una moneta), il sistema si mette alla ricerca della prima lampada spenta (il valore di 0 nel vettore testimonia questo fatto). Appena trovata, il suo valore, in array, è posto immediatamente a 6000 (che corrispondono a circa 1 minuto di accensione). Quindi esso è continuamente decrementato di 1 ad ogni centesimo di secondo ed al raggiungimento del valore 0, la relativa lampada si spegne. Lo stesso algoritmo è applicato a tutte le 24 utenze. Se si desidera un tempo maggiore di accensione è sufficiente modificare la riga che contiene il valore di 6000.

L'accensione delle 24 lampade avviene attraverso 24 condizioni booleane che determinano direttamente lo stato logico della porta. Si è preferita questa strada poiché le tre porte utilizzate (PORTB, PORTC e PORTD) sono tra loro indipendenti e non permettono adozioni di procedure unificate.

LISTATO 1 Download

```
rem ****************************
rem *     Candelabro Votivo     *
rem *                           *
rem *  by Giovanni Di Maria     *
rem ****************************

program Candelabro

dim vettore as word[24]
dim k as byte

main:

   rem Imposta funzionamento porte I/O
   trisa=1
   trisb=0
   trisc=0
   trisd=0

   rem Azzera tutte le porte
   porta=0
   portb=0
   portc=0
   portd=0

   ANSEL   =    0     ' Set AN pins to Digital I/O
   ANSELH  =    0

   rem Azzera Vettore a 24 posizioni
   for k=0 to 23
       vettore[k]=0
   next k

   while true
         rem Se si preme PULSANTE
         if porta.0 then
            while porta.0 = 1
                  delay_ms(100)
                  nop
            wend
            rem Cerca la PRIMA lampada SPENTA
            for k=0 to 23
                if vettore[k]=0 then
                   vettore[k]=6000
                   break
                end if
            next k
         end if

         rem Accende o spegne LAMPADE
         portb.0 = vettore[0] > 0
         portb.1 = vettore[1] > 0
         portb.2 = vettore[2] > 0
         portb.3 = vettore[3] > 0
         portb.4 = vettore[4] > 0
         portb.5 = vettore[5] > 0
         portb.6 = vettore[6] > 0
         portb.7 = vettore[7] > 0
         portc.0 = vettore[8] > 0
         portc.1 = vettore[9] > 0
         portc.2 = vettore[10] > 0
         portc.3 = vettore[11] > 0
         portc.4 = vettore[12] > 0
         portc.5 = vettore[13] > 0
         portc.6 = vettore[14] > 0
         portc.7 = vettore[15] > 0
         portd.0 = vettore[16] > 0
         portd.1 = vettore[17] > 0
         portd.2 = vettore[18] > 0
         portd.3 = vettore[19] > 0
         portd.4 = vettore[20] > 0
         portd.5 = vettore[21] > 0
         portd.6 = vettore[22] > 0
         portd.7 = vettore[23] > 0

         rem Decrementa secondi di tutte le lampade
         for k=0 to 23
             if vettore[k] > 0 then
                vettore[k] = vettore[k] - 1
             end if
         next k
         delay_ms(10)
   wend
end.
```

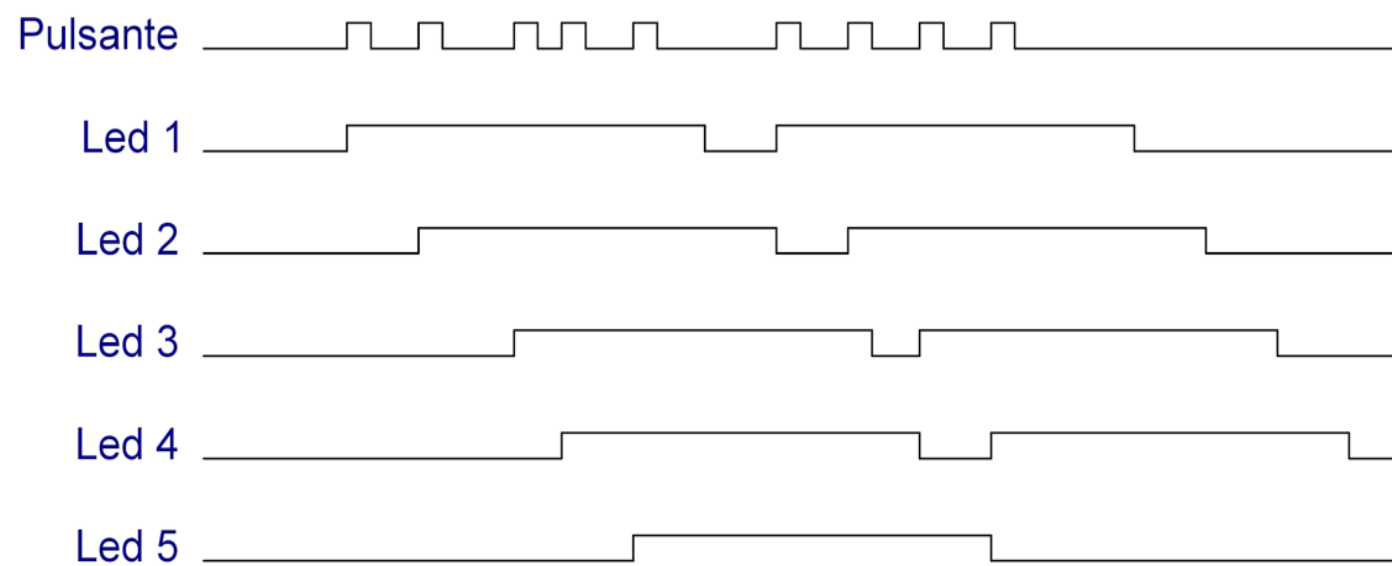



Figura 3: Grafico logico-temporale degli ingressi/uscite

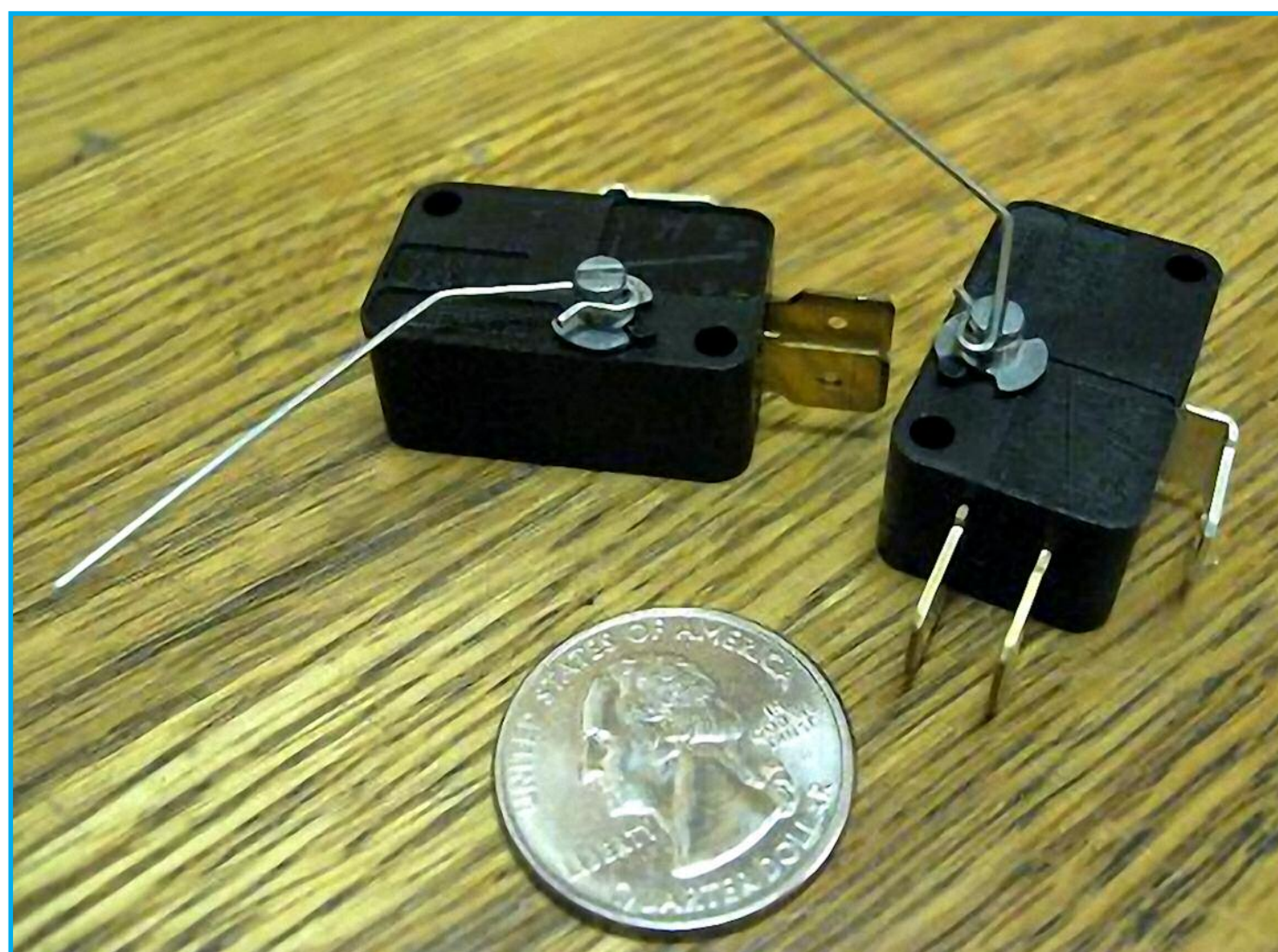



Figura 4: Un classico pulsante a moneta (coin switch)

UTILIZZI DIVERSI

Il prototipo presentato in queste pagine prevede un semplice pulsante per comandare le uscite. Le esigenze dei progettisti, ovviamente, possono modificare la tipologia dell'ingresso con altre disponibili sul mercato. Una di queste potrebbe essere costituita da un classico pulsante a moneta. Nella sua struttura base esso attua un contatto elettrico momentaneo, dovuto al peso della moneta, chiudendo per un breve periodo il circuito. Il circuito naturalmente potrebbe essere utilizzato per altri scopi, oltre alla finalità prevista dal candelabro votivo. Un esempio potrebbe essere quello di allestire una illuminazione "a tempo" per tante cabine con telefono o per tanti W.C. Chi entra e preme il pulsante determina l'accensione temporizzata della prima utenza libera. O addirittura potrebbe servire per allestire un insieme di 24 bromografi indipendenti, pilotabili con un unico pulsante.

Elenco componenti	
R1, R23	10 K 1/4 W
R2÷R22, R24÷R26	330 1/4 W
C1, C2	22pF ceramico
U2	PIC16F887
D1÷D2	Diodi Led
X1	Quarzo da 20 Mhz

Insomma, con la fantasia può essere trovata la giusta collocazione del progetto.

CONCLUSIONI

Abbiamo esaminato un utile circuito, sia dal punto di vista pratico che dall'aspetto didattico. Ricordate di allestire un circuito composto da relè e transistor di pilotaggio, semmai il prototipo venisse usato con dei carichi più robusti. Lo schema elettrico infatti prevede i diodi Led e relative resistenze di limitazione, ma il lettore può effettuare qualsiasi aggiunta utile.
Buon lavoro.

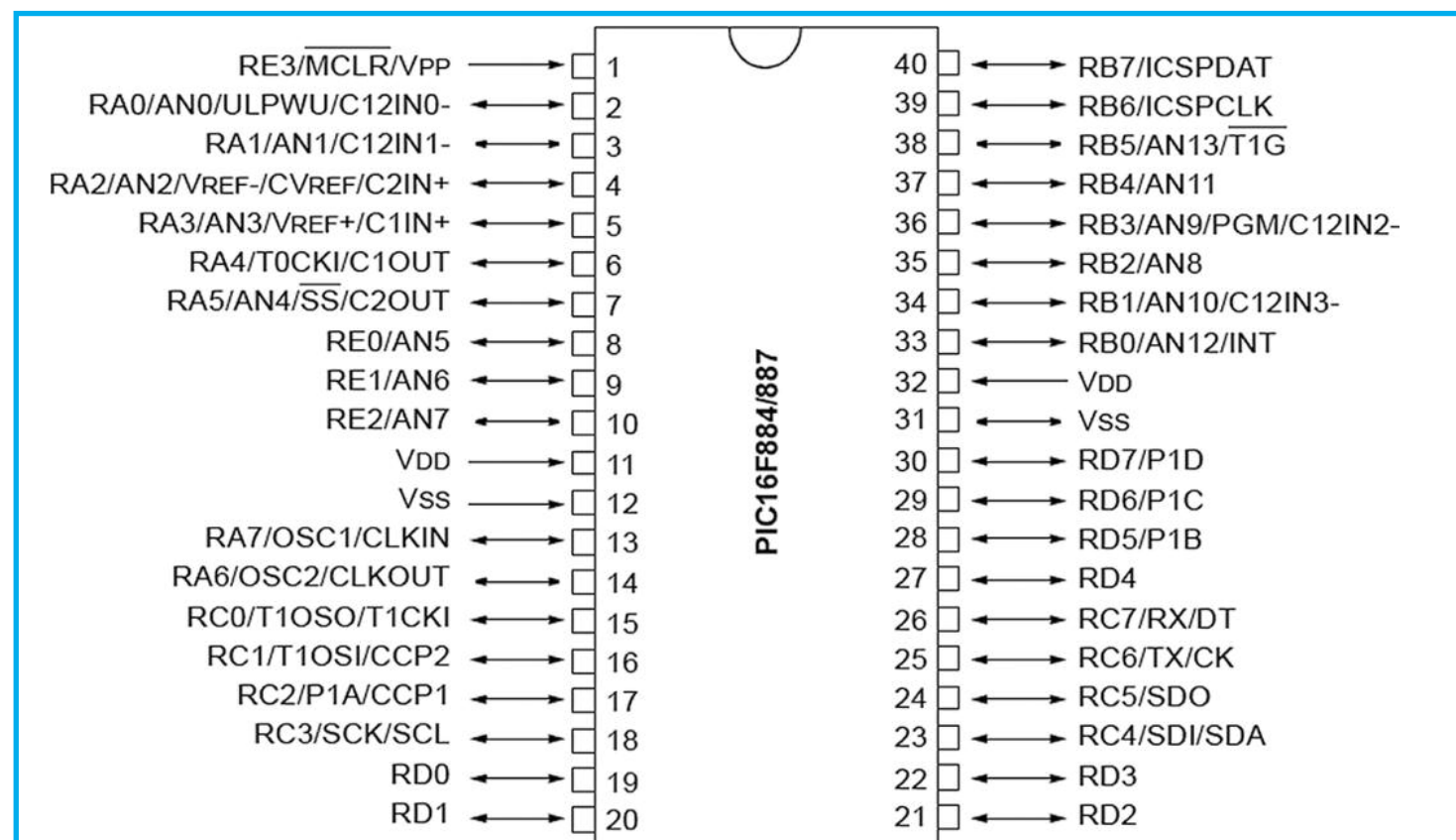



Figura 5: Pinout del Pic 16F887

progetti

di GASPARE SANTAERA

DA NOTEBOOK A BROMOGRAFO

Se avete un vecchio notebook potrete facilmente trasformarlo in un bromografo a doppia faccia. Ecco come fare

Chiunque si sia mai cimentato nello sviluppo di pcb in casa, conosce bene, il ruolo di fondamentale importanza svolto dal bromografo. Tuttavia un bromografo professionale costa un bel po', tanto da portare l'eventuale spesa al di fuori dell'amatoriale, per questo motivo tutti nel corso degl'anni ci siamo cimentati nella costruzione di bromografi più o meno professionali. Ad esempio qualche tempo fa, riciclai un vecchio progetto che girava in rete, costruendo un bromografo utilizzando il case di un vecchio scanner.
Questo progetto nasce dall'osservazione della luce prodotta dalla lampada di un banalissimo display lcd. La sorgente luminosa di quest'ultima, infatti, è abbastanza

Figura 1: Bromografo a doppia faccia da notebook

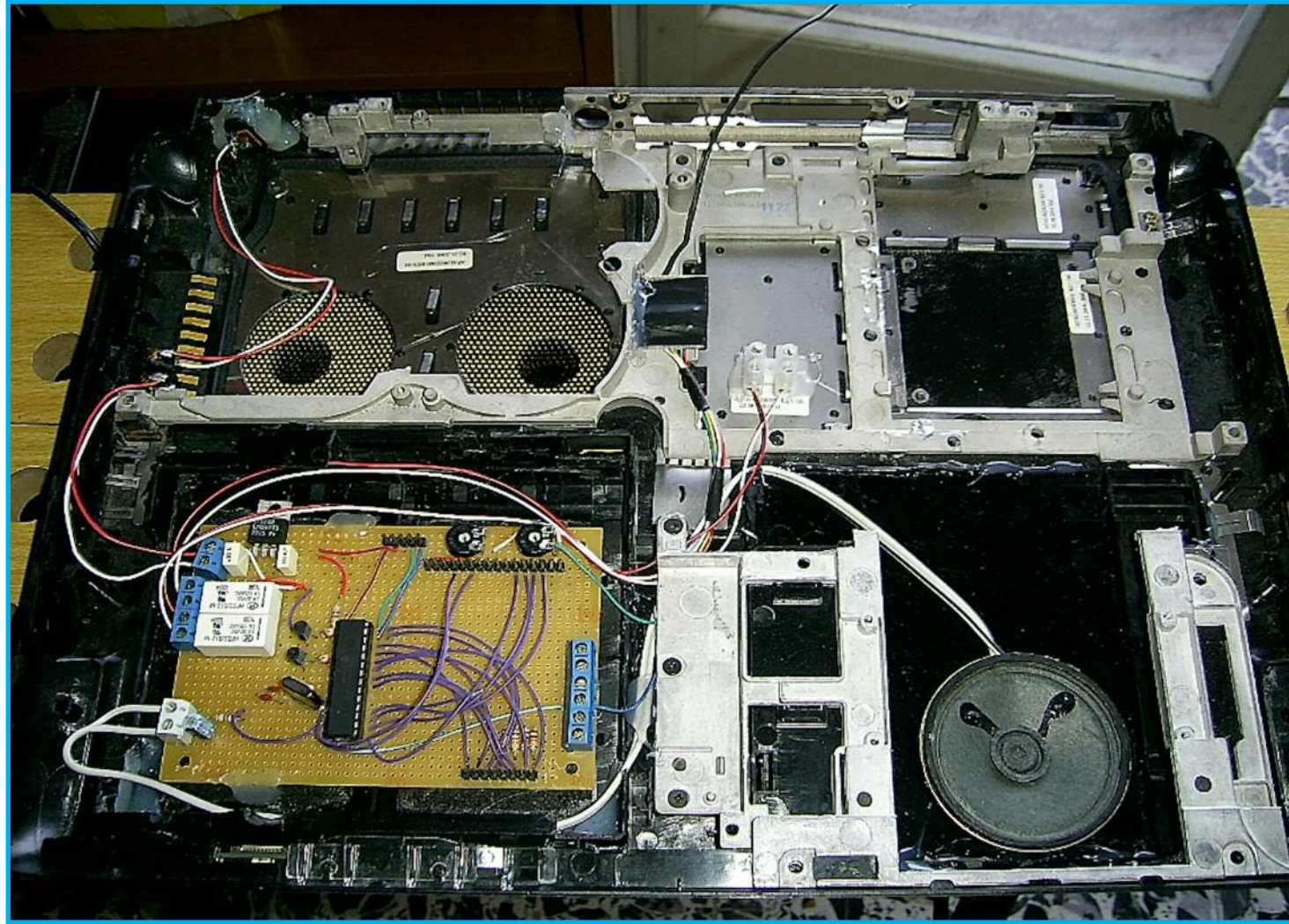



Figura 2: Parte Inferiore del Case

omogenea, requisito fondamentale per un buon bromografo, e abbastanza luminosa da impressionare il photoresist di una basetta, quindi basta riciclare un vecchio display magari recuperandone uno spaccato e il gioco è fatto.
Ed è proprio quello che ho fatto riciclando due vecchi display spaccati e un vecchio notebook, costruendo così un bromografo ad doppia faccia, compatto, economico ed efficiente.
Per quanto riguarda il notebook ho riciclato utilizzato un vecchio Toshiba con un display da 15.4 pollici, utilizzato come lampada superiore del bromografo, e un secondo display più piccolo di 10 pollici. Inoltre ho utilizzato un pic 16F876 per la realizzazione del timer e il controllo dei due display. Infine utilizzando un tastierino numerico e un display a 16 cifre per riga ho creato un piccolo sistema operativo per settaggio del timer e il numero dei display da accendere.

COSTRUZIONE

La costruzione è molto semplice, nelle figure sottostanti vengono riportati alcuni dei passaggi nonché il montaggio finale.
Per prima cosa si svuota il notebook, lasciando nel case soltanto la lampada del display principale, poi nella parte inferiore di quest'ultimo si inseriscono la scheda di controllo e uno speaker che ci avviserà con un segnale acustico della scadenza del timer. Infine sulla placca superiore viene montata la seconda lampada e il display per il controllo.

ELETTRONICA

Il cuore del bromografo è costituito dal microcontrollore, un PIC 16F876, il quale si occupa di implementare il timer, di gestire i due display e l'interfaccia utente tramite il controllo del tastierino numerico e il display alfanumerico.
Per quanto riguarda il controllo dei due display luminosi, ovvero delle due lampade, la più grande l'ho smontata da un display di un notebook da 15,4 pollici, e viene controllata tramite il suo stesso inverter, a quest'ultimo bisogna inviare oltre all'alimentazione, fissata da me a 12V, un segnale di enable di 5V ed un segnale analogico di controllo che in questo progetto viene generato dalla periferica PWM del microcontrollore e fornisce all'integrato montato sull'inverter che controlla la lampada una tensione di picco di 1.8V con una frequenza di 500hz.
Ovvero riassumendo, ho smontato l'inverter che pilotava la lampada e ho studiato il componente che gestisce il suddetto inverter, individuando i segnali e i livelli di tensione da inviare a quest'ultimo al fine di ottimizzare il controllo della lampada.

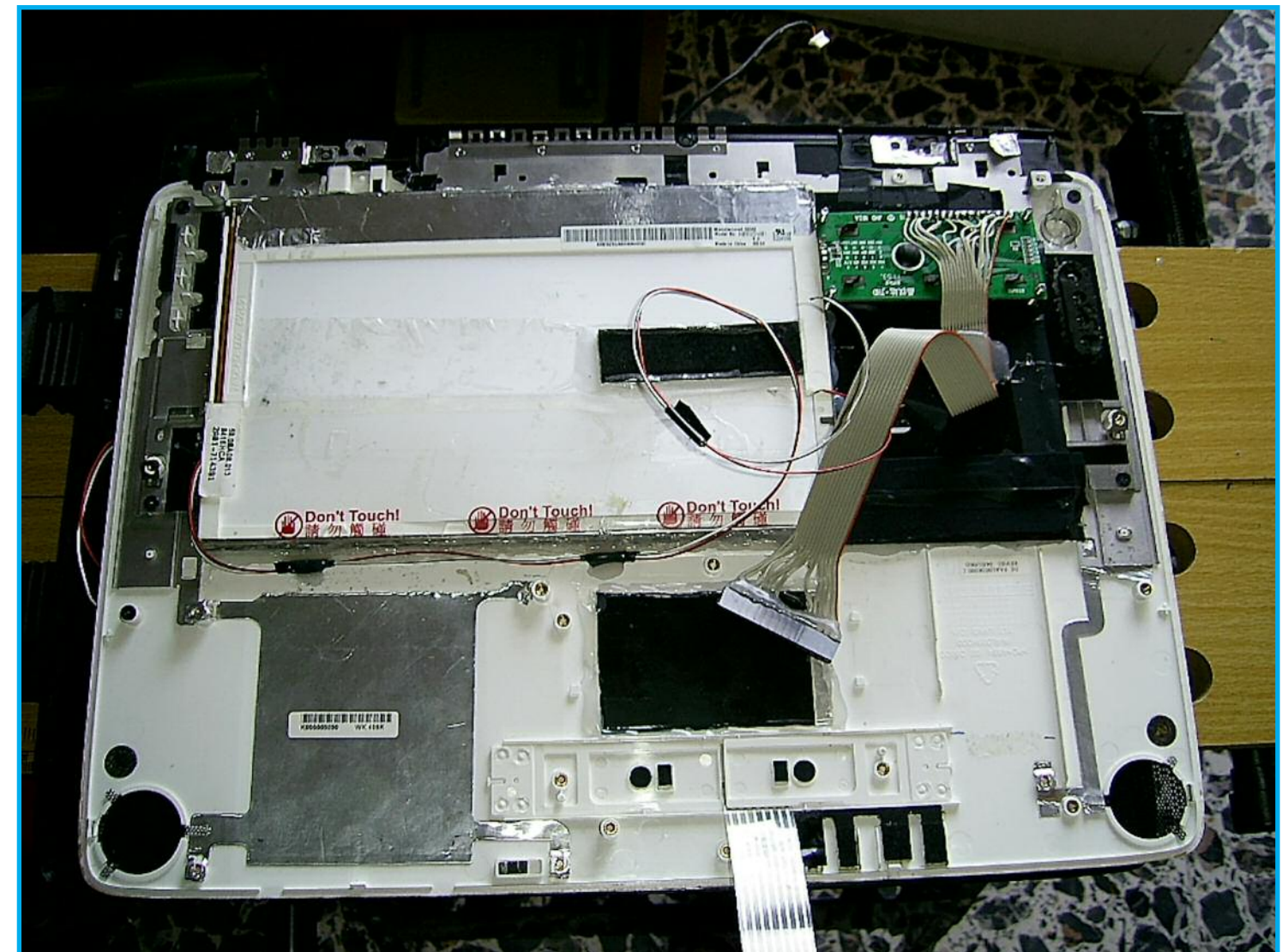



Figura 3: Placca Superiore del Notebook (Retro)

Figura 4: Schema Elettrico

Le tensioni ausiliari mandate all'integrato vengono generate direttamente dal pic mentre l'alimentazione viene fornita all'inverter tramite un relè a sua volta controllato dal pic.
Il secondo display luminoso, anche questo è stato smontato da un mini pc, ed ha un formato di 10 pollici, quest'ultimo viene controllato direttamente come se fosse una banale lampadina, basta quindi solamente alimentarlo, anche in questo caso ho utilizzato una tensione di alimentazione di 12V tramite un relè controllato dal pic.
Infine per quanto riguarda l'interfaccia utente, costituita dal display alfanumerico e dal tastierino, questa viene controllata dal pic, e permette all'utente, semplicemente seguendo le istruzioni via via riportate sul display alfanumerico di impostare il timer del bromografo e di scegliere quanti display accendere durante il processo.
Il microcontrollore una volta impostato il valore del timer e il numero di display da accendere si occupa soltanto di accendere i display tramite i due relè e di contare i secondi, alla scadenza del timer, spegne i display, resetta tutte le variabili del sistema e dopo aver eseguito un piccolo segnale acustico, tramite la sua seconda periferica PWM collegata allo speaker, e dunque pronto per un nuovo processo.

FIRMWARE

Il firmware è stato scritto in mikroC, il microcontrollore dopo una prima fase iniziale dove inizializza la porte del microcontrollore, le variabili del sistema e abilita i due segnali di controllo per l'inverter della lampada maggiore, entra in un ciclo *while* infinito, dove gestisce tramite il display e il tastierino le impostazioni utente.
L'utente viene dapprima invitato a settare il valore desiderato per il timer, e successivamente invitato a decidere se vuole utilizzare un solo display o entrambi.
Fatta quest'ultima scelta, invita l'utente a chiudere il coperchio del notebook, accende i relè interessati e si limita a far scorrere il timer semplicemente inserendo all'interno dell'esecuzione del programma un delay di durata pari in secondi al valore impostato del timer. Alla scadenza del delay e quindi del timer, spegne i relè, ovvero i display e variando la frequenza della sua seconda periferica PWM, connessa allo speaker, emette un segnale acustico composto da una serie di note musicali ascendenti (DO, RE, MI ....).

CONCLUSIONI

Il circuito visto in questa breve descrizione, mostra come riciclando alcuni semplici elementi, magari rotti come i due display, si possa creare un buon bromografo a doppia faccia, in poco tempo e sicuramente spendendo meno di quanto ne costi uno professionale.
La costruzione richiede magari un po' di pazienza soprattutto dove, come in questo caso bisogna andare a spulciare il funzionamento dell'inverter che controlla la lampada, ma onestamente visto il risultato finale, credo non ci sia niente che un po' di buona volontà non riesca a superare.

di Vincenzo Germano

SVILUPPARE E REALIZZARE PCB

Guida step by step alla realizzazione a basso costo di PCB mediante una tecnica d'incisione delle schede ramate

Tutte le volte che si parla di progetti elettronci implicitamente si parla della realizzazione di schede elettroniche comunemente conosciute come PCB (Printed Circuit Board), che siano esse prototipi o prodotti da industrializzare, in ongi caso arriva sempre il momento di passare dalle teorie, dai calcoli di progetto, dagli schematici a realizzare fisicamente il progetto. Questa guida illustrerà passo passo come arrivare a incidere PCB mediamente complessi con una tecnica a bassissimo costo.

Nella trattazione verrà presa in considerazione, come semplice esempio, una scheda ponte con due sensori magnetici (complessivamente triassiali) della Honeywell, di cui HMC1001 a singolo asse in tecnogia THT (Through Hole Tecnogoly) e HMC1002 a due assi in tecnologia SMT (Surface Mount Technology).

Questo primo articolo servirà a introdurre la tecnica di realizzazione PCB che poi verrà utilizzata in articoli successivi per progetti più complessi, come per esempio schede di prototipazione per microcontrollori quali i PIC, permettendo di autocostruirsi demo board per lo sviluppo, il debug e il testing del proprio firmware.

Ma questo lo vedremo nella prossima puntata.

Figura 1: Layout della scheda con i sensori Honeywell

PANORAMICA SU ALCUNE TECNICHE DISPONIBILI, VANTAGGI/SVANTAGGI:

Generalmente dopo aver progettato e testato, ad esempio mediante "Breadboard" il proprio progetto, il passagio successivo è quello di scegliere un supporto sul quale saldare i componenti e molto spesso si opta per soluzioni a basso costo come schede millefori per evitare di mandare il

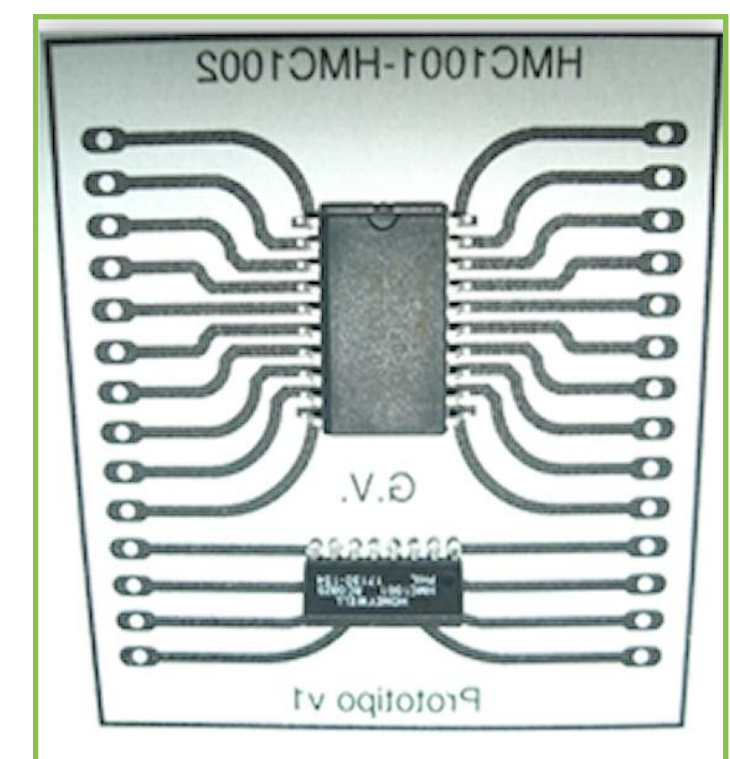



Figura 2: Confronto con stampa

progetto in fonderia e spendere cifre molto più alte. Tali schede però hanno il vincolo di permettere il facile montaggio di componenti THT ma risultano molto scomode o addirittura inutilizzabili per comonenti in SMT; in più bisogna tracciare le piste con stagno o fili per effettuare le connessioni, che in progetti più o meno complessi risulta di difficile attuazione; infine le schede realizzate con le basi millefori sono pezzi unici e nel caso in cui si vuole modificare o realizzare un altro prototipo dello stesso progetto, bisogna rifarle da zero.
Una soluzione intermedia usata è il trasferimento diretto, nel quale si disegna a mano le piste su basette ramate, mediante pennarelli indelebili e poi si passa a inciderle come vedremo nella tecnica successivamente trattata ma anche in questo caso ogni nuovo progetto (o modifica) bisogna ridisegnarlo, con conseguente perdita di tempo. In più con componenti SMD (Surface Mount Device) non si riuscirebbe a ottenere *footprint* complessi, dovendoli disegnare a mano.
Un'ottima soluzione tra le varie tecniche di realizzazione PCB è l'utilizzo del bromografo e di basette pre-sensibilizzate (mediante uno strato di vernice fotosensibile) ma una scelta di questo tipo richiede comunque un dispositivo che è appunto il bromografo. I vantaggi sono molteplici, dalla ripetibilità delle schede mediante lo stesso layout allo sbroglio di circuiti molto più complessi (anche con doppio layer) ma tutto ciò presuppone il bromografo e il reperimento di basette pre-sensibilizzate, con costi maggiori rispetto alle tecniche sopracitate.
Infine una soluzione di compromesso riguarda l'utilizzo di fogli "*Press'n'Peel*". Il metodo di seguito illustrato si avvicina a quello dei fogli "*Press'n'Peel*", con i vantaggi rispetto ai precedenti di essere molto veloce, poco costoso, applicabile facilmente senza strumenti sofisticati, come può essere un bromografo.
La tecnica che verrà illustrata non è orientata alla progettazione/sbroglio del progetto ma solo alla realizzazione del PCB, partendo dal presupposto di essere già in possesso dello schematico sbrogliato (layout scheda).

Figura 3: Stampa su foglio lucido

STEP 1: STAMPA E VERIFICA

Una volta in possesso del layout della scheda, nel caso in questione quella dei due sensori Honeywell visibile in Figura 1, è consigliabile stampare il master (stampa 1:1) per verificare effettivamente se i *footprint* corrispondono con i package dei componenti, magari sovrapponendo i componenti (più critici es integrati) sulla stampa, come si nota in Figura 2, nella quale si vedono i sensori HMC1001 e HMC1002 sul foglio stampato.
In seguito il master deve essere stampato necessariamente con una stampante "*laser*" su un particolare foglio che deve soddisfare le seguenti caratteristiche: deve essere lucido per evitare di assorbire il toner della stampante, abbastanza sottile e soprattutto si deve "*disfare*" in acqua. A tale scopo, al posto dei fogli "*Press'n'Peel*" sono state fatte delle prove con dei sepmlici giornali; trovato quello che soddisfa i requisiti sopra indicati, su di esso è stato stampato il circuito come riportato in Figura 3. La stampa deve essere fatta specchiata rispetto all'originale per permettere di avere il circuito nel verso giusto una volta riportato sulla scheda ramata.
Viene utilizzata una stampante laser perché il suo toner plastico è resistente alla soluzione chimica utilizzata in seguito, perciò una volta che la basetta con la stampa impressa verrà attaccata chimicamente, la parte sottostante il toner, non verrà erosa ma il tutto sarà spiegato e più chiaro in seguito.

STEP 2: PULIZIA SCHEDA RAMATA

Il passo successivo è quello di ripulire la basetta mediante lana d'acciaio finissima; questo passaggio serve a togliere le impurità che ci sono sulla scheda e nel caso in cui non venga effettuato, l'intero processo viene compromesso. Come si evince dalla Figura 4 il rame sulla scheda risulta molto lucido e indica che sono state eliminate

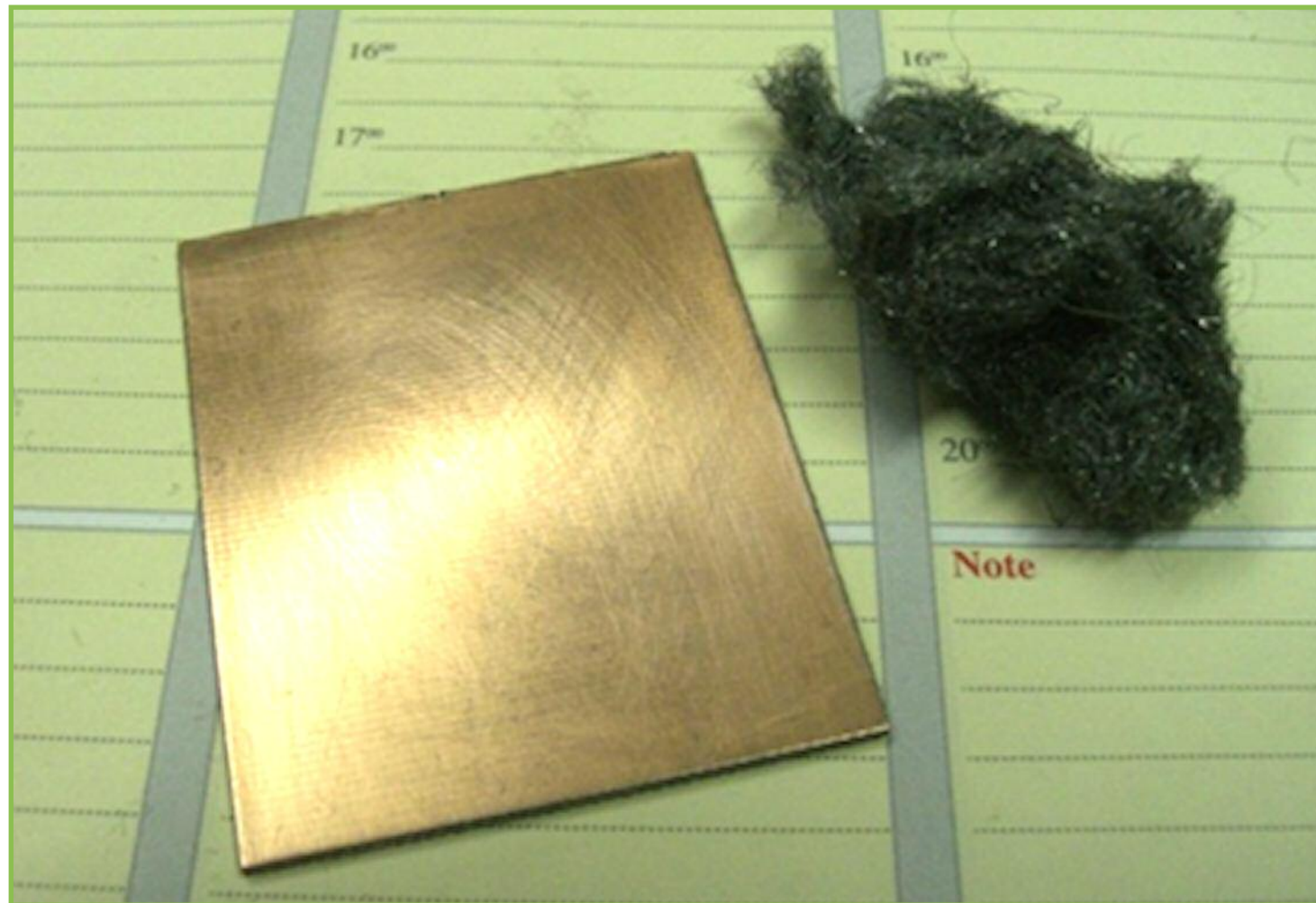



Figura 4: Pulizia scheda ramata

tutte le parti ossidate. Questo passaggio è da effettuare poco prima del prossimo, perché se passa troppo tempo il rame potrebbe ossidarsi ancora e perciò bisognerebbe pulirlo con conseguente degrado del PCB.
Si faccia attenzione a non strofinare troppo in profondità o a non ripulirlo troppe volte, altrimenti il sottile strato di rame depositato sulla basetta potrebbe consumarsi totalmente, rendendo la basetta inutilizzabile.

STEP 3: STAMPA SU BASETTA RAMATA

A questo punto bisogna far riscaldare un comune ferro da stiro,senza l'opzione del vapore e si può usare un qualsiasi ferro da stiro anche vecchio e in disuso, l'importante che la sua temperatura raggiunga i gradi necessai per il nostro scopo.
Lo si poggia sulla basetta appena pulita e la si riscalda per circa 30/40 secondi. Fate attenzione che nel caso in cui il calore del ferro è troppo alto dopo poco il rame del PCB cambia colore diventando di un arancione differente in alcuni punti, segno che la temperatura è troppo alta, perciò bisogna abbassarla per le prossime volte. Dopo averlo riscaldato si poggia sulla scheda il foglio stampato sulla carta lucida, con il lato che presenta l'inchiostro plastico appoggiato sul rame caldo, prestando attenzione a centrarla bene, altrimenti se parti delle piste vanno fuori dalla basetta bisogna rifare tutto il processo dallo step 1. Si tiene premuto il ferro da stiro sul foglio per circa un minuto, cercando di schiacciare bene su tutte le parti Figura 5.
Per il prossimo passo si possono seguire due strade: si fa raffreddare la basetta e senza staccare la carta da essa si immerge il tutto in acqua fredda Figura 6; oppure con la scheda ancora calda, si stacca la carta da essa, facendo attenzione a non far saltare le piste e a non scottarsi (tenendo presente che la basetta è "bollente"). La procedura migliore e da me più utilizzata è la prima.
Perciò si lascia la basetta in acqua per circa 10 minuti, dopodiché delicatamente si toglie tutta la carta, come mostrato in Figura 7, dalla quale si vede anche che ci sono alcune imperfezioni, come: piste saltate o pad non definiti. Per rimediare si può utilizzare un comune e classico pennarello nero indelebile Figura 7.

STEP 4: INCISIONE

Per la realizzazione di questa scheda, la basetta utilizzata è di vetronite e non bachelite; è stata scelta per le sue migliori caratteristiche, tra le più importanti ricordiamo: buon isolamento, l'alta resistenza alle sollecitazioni meccaniche, stabilità dimensionale nel tempo e in più presenta una buona resistenza agli shock termici di breve durata ma non nel lungo periodo.
A questo punto non rimane altro che fare l'incisione chimica mediante il "*Percloruro Ferrico*" (da maneggiare con molta attenzione e le dovute protezioni, utilizzando solo recipienti e pinzette di plastica). É reperibile facilmente sia liquido che solido da

Figura 5: Stampa su scheda ramata

Figura 6: Immersione scheda in acqua

trattare e successivamente va smaltito seguendo le dovute accortezze.
Il percloruro andrà a eliminare tutto il rame in eccesso che non si trova sotto le piste stampate.
La scheda impiega circa 30 minuti per essere incisa Figura 8, anche se il tempo varia a seconda delle sue dimensioni (più grande è la dimensione della basetta più tempo ci vuole) ma dipende anche dalla temperatura dell'agente chimico; infatti se si riscalda il recipiente con il percloruro,

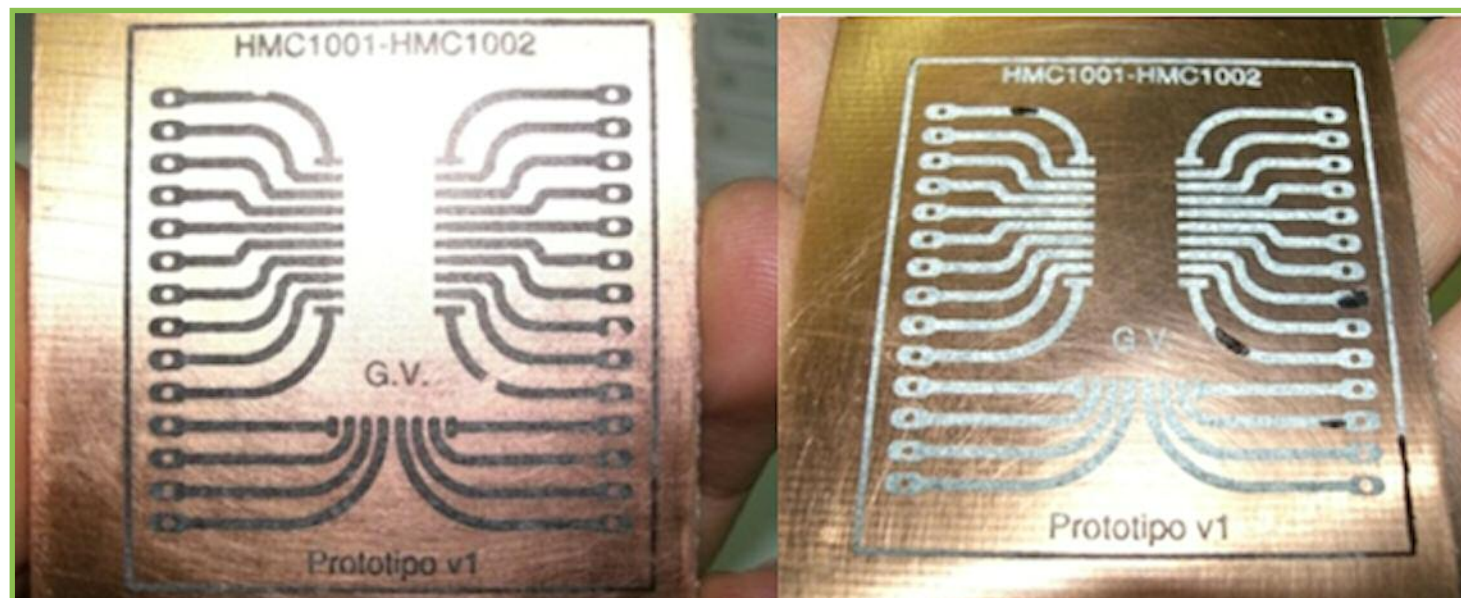



Figura 7: Scheda con imperfezioni - Scheda ritoccata con il pennarello

Figura 8: Immersione nel percloruro ferrico

davanti a una fonte di calore (con la scheda in immersione) e lo si agita delicatamente i tempi si riducono notevolmente (attenzione a non respirare i fumi che rilascia il percloruro nel caso si riscaldi troppo). È consigliabile controllare costantemente la basetta (tirandola fuori dall'agente chimico) per vedere quando tutto il rame in eccesso si è tolto, perché se rimane troppo a lungo si potrebbero rovinare le piste, dato che il percloruro comincerebbe a "*corrodere*" i bordi.
Successivamente la basetta va lavata abbondantemente sotto acqua corrente.

STEP 4: PULIZIA E FORATURA

L'ultimo passaggio prima della foratura è l'eliminazione del toner plastico dalle piste ramate. In principio è stato utilizzato l'acetone per le unghie, che però permetteva al toner di togliersi dalle piste e finire sulla basetta (sul materiale isolante), con un conseguente inestetismo ma senza nessun problema di carattere pratico; successivamente però si è riutilizzata la lana d'acciaio fine, facendo però attenzione a non danneggiare le piste. I risultati si possono vedere nella Figura 9, in cui si nota la basetta con ancora il toner sulle piste e dopo il passaggio con lana d'acciaio.
Il procedimento per la realizzazione della PCB è terminato. I passi che sono stati effettuati successivamente sono: taglio dei bordi (mediante seghetto o taglierino), foratura (mediante trapano) e saldatura componenti.
Per la foratura è stato utilizzato un trapano con una punta da 0,8 mm, riportato il dettaglio in Figura 10; sono state effettuate anche prove con una punta da 0,5 mm ma con scarso successo vista la fragilità di tali punte. In Figura 10 è riportata la scheda dopo la foratura relativa al sensore HMC1001; mentre nella Figura 11 viene riportata, da due angolazioni differenti, la "Printed Circuit Board" completa dopo la saldatura componenti.
Oltre ai sensori sono stati saldati anche

Figura 9: Scheda prima della pulizia - Scheda dopo la pulizia

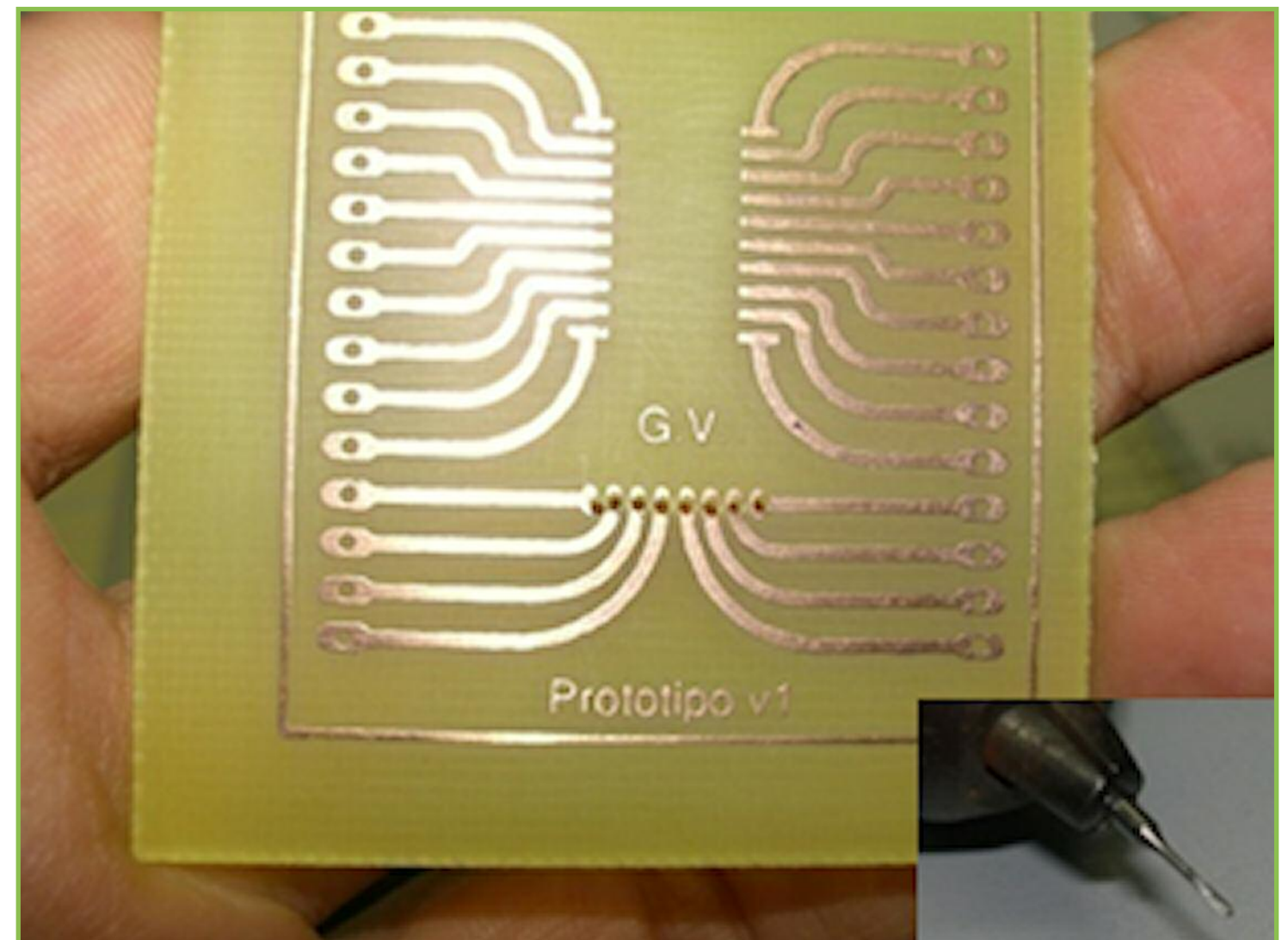



Figura 10: Scheda dopo la foratura - Dettaglio punta da 0,8 mm

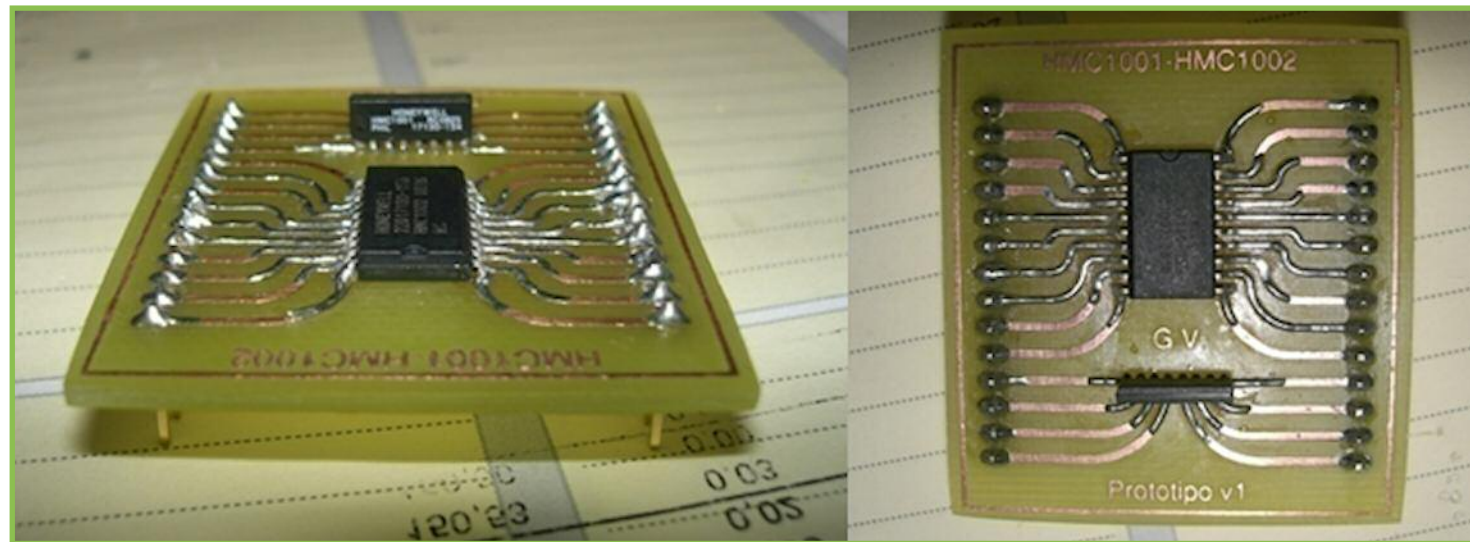



Figura 11: PCB dei sensori terminata dopo la saldatura componenti

due "*hader*" per poterla collegare come "*shield*" su altre schede.

CONCLUSIONI:

Mediante la tecnica illustrata è possibile realizzare PCB mediamente complessi a costi relativamente bassi, considerando l'acquisto del percloruro ferrico e delle bastte ramate, il resto è tutto reperibile praticamente a costo nullo. Il solo presupposto è quello di avere il layout della scheda. Con questa procedura nel prossimo articolo sarà realizzata una demodoard per PIC, che permetterà di programmarli/debuggarli e in più avrà integrata on-board una millefori per la saldatura di componenti aggiuntivi.

Sensori Honeywell HMC100x

I sensori magnetici HMC1001/1002, utilizzano la tecnologia AMR (Anisotropic Magnetoresistive) e sono rispettivamente a uno e due assi. Trovano applicazioni in molti campi come: compassi, sistemi di navigazione, magnetometria, strumenti medici, misure di corrente, misure di intensità del campo magnetico terrestre (direzione) e molto altro ancora. Sono estremamente sensibili, percepiscono induzioni magnetiche molto piccole, da circa 10 gauss fino a 6 gauss, possono lavorare con tensioni di alimentazione piccole (2 V), sono compatibili con sistemi alimentati a batterie e sono forniti di due caratteristiche aggiuntive rispetto ad altri sensori che sono: la gestione dell'offset e il set/reset. Altre caratteristiche sono: misura di direzione e intensità dell'induzione magnetica; identificazione di variazioni (di campo) dovute a materiali ferromagnetici; bassa isteresi e alta linearità; non necessitano di nessun solenoide esterno per il set/reset.

Questi due sensori (AMR) sfruttano il concetto del ponte di "Wheatstone" per effettuare la misura dell'induzione magnetica, perciò necessitano esclusivamente di una tensione di alimentazione e grazie a essa convertono l'induzione magnetica incidente (lungo la direzione dell'asse sensibile, "sensitive axis") in un valore differenziale di tensione, che indica sia l'intensità del campo che la direzione.

di ANTONIO GIANNICO

CORSO MIKROPASCAL PER PIC

PILOTARE LCD GRAFICI (GLCD)

(parte undicesima)

Gli LCD Grafici (GLCD-Graphic Liquid Crystal Display) sono terminali di visualizzazione impiegati come interfaccia utente in moltissime apparecchiature elettroniche. In questa puntata del corso mostreremo come interfacciare fisicamente display di questo tipo con un controllore PIC e come pilotarli utilizzando codice scritto in linguaggio MikroPascal

Figura 1-GLCD 128x64 KS0108 presente sulla scheda di sviluppo EasyPIC3 [2]. Il pin-out e l'interfacciamento hardware e firmware è analogo a quello di altri GLCD altrettanto diffusi aventi la stessa risoluzione o risoluzione differente come i T6963 128x128, 240x64 e 240x128 oppure il Nokia 3310.

Nelle puntate 5 e 6 del corso abbiamo parlato di display LCD alfanumerici; in particolare abbiamo mostrato come interfacciare display di questo tipo con un controllore PIC Microchip e come pilotarlo attraverso l'impiego di codice MikroPascal. Abbiamo così scoperto che MikroPascal mette a disposizione una serie di routine di libreria che rendono semplice e immediato il controllo ad alto livello di questo tipo di visualizzatore. Qualcosa di analogo vale anche per i display *LCD grafici* anche noti come *GLCD* (***G**raphic **L**iquid **C**rystal **D**isplay*). A questi è dedicata la presente lezione del corso il cui scopo è quello di mostrare nell'ordine:

- quali sono le caratteristiche principali di un display LCD grafico e quale il suo funzionamento;

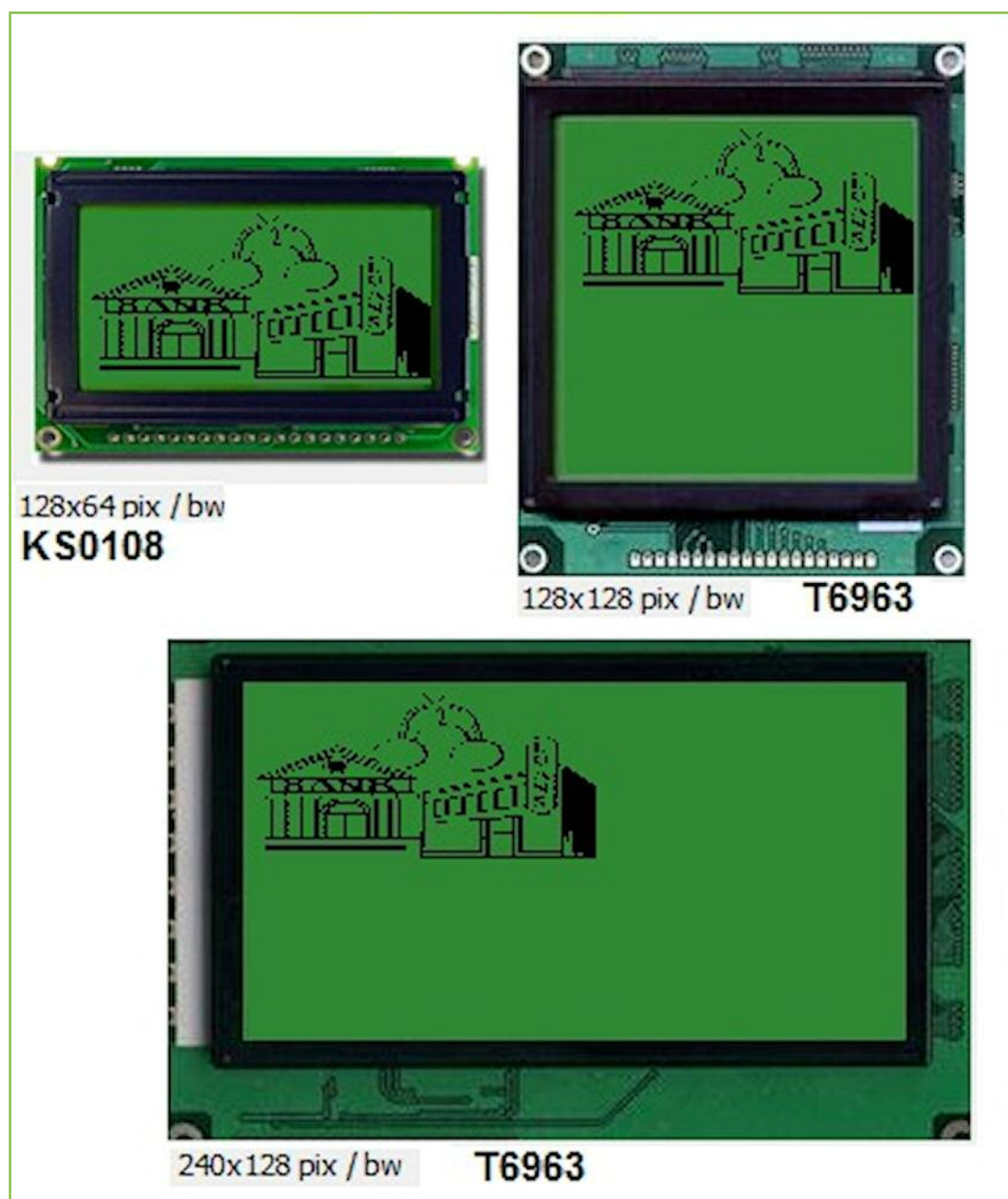



Figura 2-Limmagine mostra come appare una stessa immagine 128x64 punti rappresentata su tre diversi display grafici: KS108 128x64, T6963 128x128 e T6963 240x128.

- in che modo è possibile connettere fisicamente questo tipo di display al controllore PIC;
- quali sono le routine di libreria MikroPascal che è possibile utilizzare per il controllo del display.

In relazione all'ultimo punto, non mancheremo di proporre, come al solito, esempi pratici o dimostrativi finalizzati ad un efficace apprendimento delle nozioni.

A differenza di quanto è possibile fare con un LCD alfanumerico, con un LCD grafico è possibile tracciare punti, linee rette o curve e addirittura riprodurre immagini sebbene monocromatiche e di dimensioni e risoluzione contenuta.

Tutto ciò è comunque sufficiente a rendere i GLCD utili strumenti di interfaccia uomo-macchina realizzabili attraverso applicazioni firmware di limitata complessità. Il parametro fondamentale che caratterizza un GLCD è la risoluzione intesa come il numero di punti che possono compore l'immagine rappresentata sulla sua superficie. Nel proseguimento della lezione faremo prevalentemente riferimento a GLCD 128x64 dove 128 è il numero di punti lungo l'asse orizzontale mentre 64 è quello disponibile lungo l'asse verticale, ma qualcosa di analogo può essere detto anche per GLCD di altro formato.

In Figura 1 è mostrato il *GLCD 128x64 KS0108* presente sulla scheda di sviluppo EasyPIC3 [2].

Analoghi GLCD sono a corredo dei successivi modelli di EasyPIC e di molte atre scede e tool di sviluppo per controllori PIC. Il pin-out e l'interfacciamento hardware e firmware è analogo a quello di altri GLCD altrettanto diffusi aventi la stessa risoluzione o risoluzione maggiore come i T6963 128x128, 240x64 e 240x128 oppure il Nokia 3310.

In Figura 2 è mostrato come appare una stessa immagine rappresentata su tre diversi display grafici: KS108 128x64, T6963 128x128 e T6963 240x128.

Le tre immagini sono tratte direttamente dal tool *GLCD Bitmap Editor* integrato all'interno dell'ambiente di sviluppo MikroPascal PRO for PIC. Di questo tool parleremo in maniera più approfondita nella prossima puntata 12 del corso MikroPascal PRO for PIC.

MikroPascal Pro for PIC, come per gli LCD alfanumerici, anche per gli LCD grafici (GLCD) mette a disposizione una serie di routine di libreria attraverso le quali è possibile interfacciare e gestire agevolmente display grafici. In particolare, MikroPascal Pro for PIC dispone di librerie "preconfezionate" per pilotare display grafici dei seguenti tipi:

- *Samsung KS0108* (risoluzione 128x64);
- *Toshiba T6963* (risoluzioni 240x128, 240x64 e 128x128);
- *Nokia 3310*.

INTERFACCIAMENTO TRA GLCD E CONTROLLORE PIC

Lo schema elettrico che definisce l'interconnessione tra controllore PIC e GLCD (Figura 3) è dettato dalle specifiche funzioni delle linee dati e di controllo di questo ultimo; il firmware è quindi stilato di conseguenza. La questione della suddivisione delle linee del display in linee di controllo e linee dati (14 linee complessive) ricalca in buona parte quanto detto nelle puntate 5 e 6 per il display alfanumerico standard Hitachi HD44780. A questi 14 pin vanno aggiunti, come al solito, i pin di alimentazione (5Vcc e GND), il pin di regolazione del contrasto (Vo) e i pin di retroilluminazione (Led+ e Led-).

In questa lezione dedicata ai GLCD ci occuperemo solamente del display KS0108 128x64 per due principali motivi: è un display reperibile facilmente e di ridotto costo e soprattutto equipaggia numerose schede e kit di sviluppo per PIC tra cui le note Easy PIC.

In Tabella 1 si riporta il pin-out di questo tipo di GLCD. Come è possibile osservare, le funzioni dei singoli pin sono analoghe a quelle che abbiamo illustrato per gli LCD

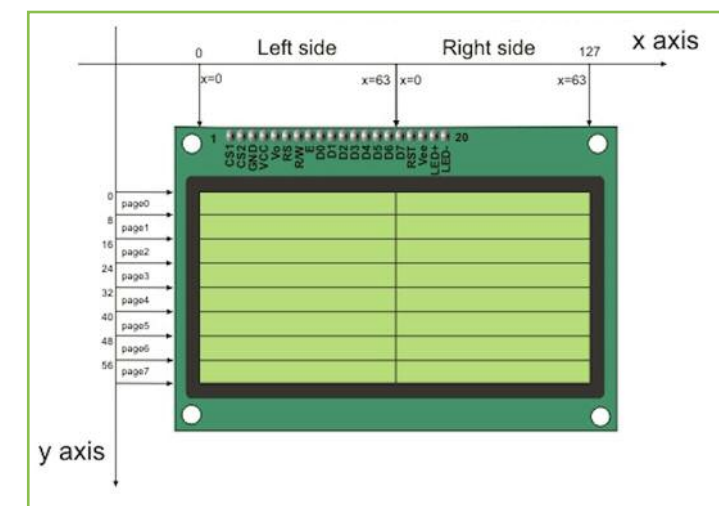



Figura 3-Schematizzazione di un display grafico KS0108 128x64 (suddivisione in banchi e pagine) [2].

Pin	Acronimo	Significato
1	CS1	Chip Select 1
2	CS2	Chi Select 2
3	GND	Power suply–(Ground)
4	Vcc	Power supply + (5V)
5	Vo	Contrast Adjust
6	RS	Register Select
7	R/W	Read/Write
8	E	Enable
9	DB0	Data bus line 0 (LSB)
10	DB1	Data bus line 1
11	DB2	Data bus line 1
12	DB3	Data bus line 3
13	DB4	Data bus line 4
14	DB5	Data bus line 5
15	DB6	Data bus line 6
16	DB7	Data bus line 7 (MSB)
17	RST	ReSeT
18	Vee	Contrast Adjust power supply
19	LED+	Led backlight (anodo)
20	LED-	Led backlight (catodo)

Tabella 1- Pin-out dei display grafici con controller KS0108

alfanumerici nel corso delle puntate numero 5 e 6.

Al fine di fugare qualunque dubbio e fissare bene i concetti è opportuno sottolineare alcuni aspetti di questo tipo di display:

- il GLCD integra sei pin di controllo (E, RS, R/W, CS1, CS2, RST);
- presenta un bus dati a 8 bit;
- presenta normalmente due pin per la retroilluminazione (Led+ e LED-);
- presenta due pin di alimentazione (Vcc=+5V e GND);
- a differenza di quanto accade per gli LCD alfanumerici possono essere pilotati dal PIC sfruttando l'intero bus dati e tutti i pin di controllo per complessivi 14 pin; non è quindi possibile un pilotaggio in modalità a 4 bit (4 bit mode).

Quanto detto, unitamente a quanto diremo successivamente sulle librerie MikroPascal è sufficiente a inquadrare in maniera esaustiva l'argomento. Ci limitiamo ad aggiungere che un driver firmware che gestisce un display di questo tipo tiene conto dell'organizzazione del display mostrata in Figura 3. La schematizzazione prevede la scomposizione in due banchi (lato destro e lato sinistro) ciascuno dei quali risulta scomposto (per un GLCD 128x64) a sua volta in otto pagine aventi ciascuna dimensione 64x8 punti.

LE LIBRERIE FIRMWARE DI MKROPASCAL PRO FOR PIC

Quando si utilizzano all'interno di un codice MikroPascal delle routine di libreria per la gestione di un GLCD, è necessario dichiarare la libreria stessa, esattamente come accade per funzioni o procedure di qualunque altra libreria.

In MikroPascal questo va fatto spuntando, all'interno della Library Manager, le voci *Glcd* e *Glcd_Fonts* (Figura 4). In realtà è sufficiente spuntare unicamente la prima delle due dal momento che la spunta sulla seconda appare di conseguenza automaticamente.

LA CORRISPONDENZA TRA FIRMWARE E HARDWARE

La connessione fisica tra controllore PIC e display deve essere rispettata anche a livello di gestione firmware. Gli esempi che mostreremo di seguito mettono in risalto il modo in cui vengono dichiarati i pin di controllo e quelli dei dati del controllore e la corrispondenza con quelli del display. In Tabella 2 riportiamo, in particolare, l'insieme delle dichiarazioni globali che è necessario inserire all'interno del codice MikroPascal.

ROUTINE DI LIBRERIA PER LA GESTIONE DI GLCD

La maggior parte delle routine di libreria servono per gestire il display in scrittura e pertanto non restituiscono valori. La maggior parte di esse necessitano per questo di parametri atti ad individuare punti o entità grafiche come linee, cerchi, rettangoli da rappresentare. MikroPascal PRO for PIC mette a disposizione per queste operazioni una serie di routine raccolte nelle librerie *GLCD* e *GLCD_Fonts*. E' comune suddividere queste routine in due categorie:

- *routine base;*
- *routine avanzate.*

Le prime verranno analizzate nei paragrafi che seguono, le seconde verranno invece analizzate parte nei paragrafi che seguono e parte nella prossima puntata del corso.

ROUTINE DI BASE (BASIC ROUTINE) PER IL CONTROLLO DEL GLCD

Tra le funzioni base per il controllo di un GLCD troviamo le seguenti:

- *Glcd_Init;*
- *Glcd_Set_Side;*
- *Glcd_Set_X;*
- *Glcd_Set_Page;*
- *Glcd_Read_Data;*
- *Glcd_Write_Data.*

Di seguito le analizziamo singolarmente al fine di comprenderne appieno il significato e il corretto impiego.

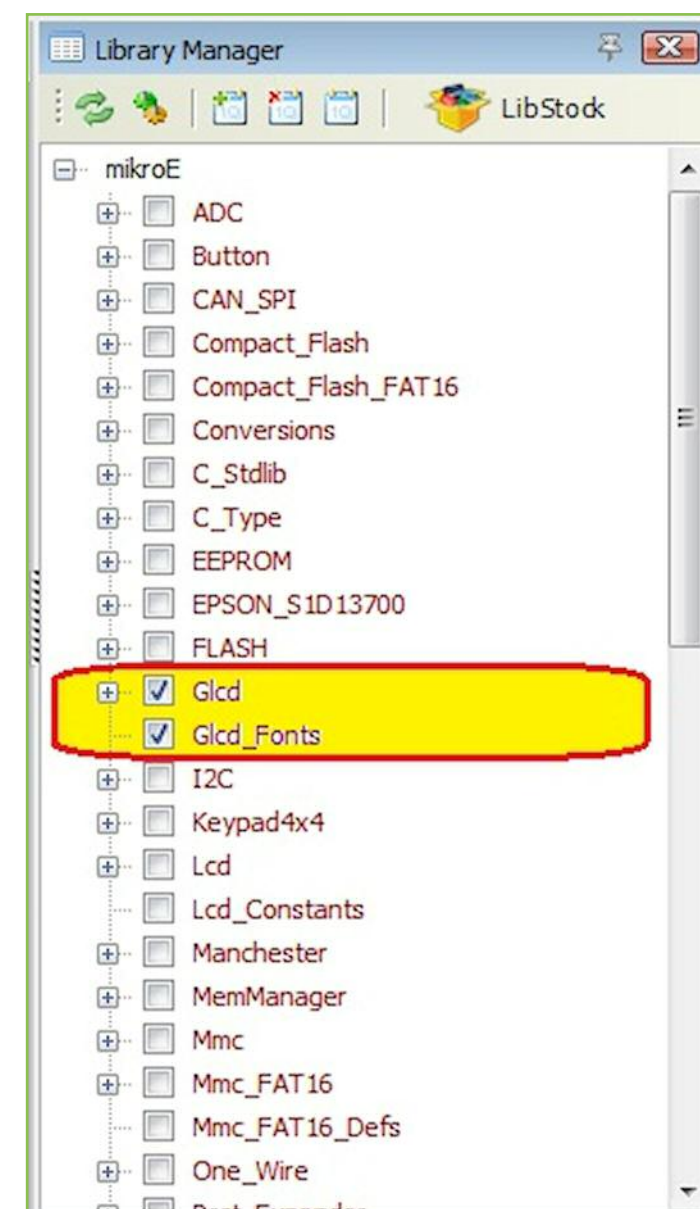



Figura 4-Library Manager di MikroPascal PRO for PIC. Si notino le spunte alle voci Glcd e Glcd_Fonts (librerie contenenti le funzioni per la gestione del GLCD).

Glcd_Init (Inizializzazione del display)

Questa routine di libreria consente di inizializzare il display grafico. Si tratta della routine equivalente alla *Lcd_Init* incontrata nel corso delle puntate 5 e 6 che hanno riguardato i display alfanumerici.
Essa deve sempre precedere qualunque altra routine di libreria utilizzata per la gestione del display GLCD. A sua volta necessita che sia stata preventivamente definita l'interconnessione fisica tra display e PIC (Tabella 2).
Si rimanda al contenuto della Tabella 3 per maggiori dettagli.

Glcd_Set_Side (Selezione di un banco o lato)

Come visto precedentemente, il display è organizzato in banchi (o lati). E' per questo spesso necessario disporre di istruzioni attraverso le quali poter selezionare il banco. Questa operazione è possibile attraverso l'impiego della routine di libreria predefinita *Glcd_Set_Side*. Si rimanda al contenuto della Tabella 4 per ulteriori dettagli.

Glcd_Set_X (Posizionamento all'interno del banco)

Così come può essere necessario puntare un banco del display, analogamente può essere necessario puntare una determinata posizione all'interno di un banco. A tale proposito è disponibile la funzione *Glcd_Set_X*, oltre a *Glcd_Set_Page* che illustreremo successivamente. Si rimanda al contenuto della Tabella 5 per maggiori dettagli.

Glcd_Set_Page (Selezione di una pagina)

Essendo il display organizzato in banchi e

Variabili da definire nel codice che impiega le librerie per il controllo del GLCD	Descrizione	Esempio tipico
var GLCD_DataPort : byte; **sfr; external;**	Glcd Data Port	**var** GLCD_DataPort : byte **at** PORTD;
var GLCD_CS1 : **sbit; sfr; external;**	Chip Select 1 line	**var** GLCD_CS1 : **sbit at** RB0_bit;
var GLCD_CS2 : **sbit; sfr; external;**	Chip Select 2 line	**var** GLCD_CS2 : **sbit at** RB1_bit;
var GLCD_RS : **sbit; sfr; external;**	Register select line.	**var** GLCD_RS : **sbit at** RB2_bit;
var GLCD_RW : **sbit; sfr; external;**	Read/Write line.	**var** GLCD_RW : **sbit at** RB3_bit;
var GLCD_EN : **sbit; sfr; external;**	Enable line.	**var** GLCD_EN : **sbit at** RB4_bit;
var GLCD_RST : **sbit; sfr; external;**	Reset line.	**var** GLCD_RST : **sbit at** RB5_bit;
var GLCD_CS1_Direction : **sbit; sfr; external;**	Direction of the Chip Select 1 pin.	**var** GLCD_CS1_Direction : **sbit at** TRISB0_bit;
var GLCD_CS2_Direction : **sbit; sfr; external;**	Direction of the Chip Select 2 pin.	**var** GLCD_CS2_Direction : **sbit at** TRISB1_bit;
var GLCD_RS_Direction : **sbit; sfr; external;**	Direction of the Register select pin.	**var** GLCD_RS_Direction : **sbit at** TRISB2_bit;
var GLCD_RW_Direction : **sbit; sfr; external;**	Direction of the Read/Write pin.	**var** GLCD_RW_Direction : **sbit at** TRISB3_bit;
var GLCD_EN_Direction : **sbit; sfr; external;**	Direction of the Enable pin.	**var** GLCD_EN_Direction : **sbit at** TRISB4_bit;
var GLCD_RST_Direction : **sbit; sfr; external;**	Direction of the Reset pin.	**var** GLCD_RST_Direction : **sbit at** TRISB5_bit;

Tabella 2-Dichiarazioni globali necessarie all'interno del codice MikroPascal per il controllo del display grafico (GLCD) KS0108 di Samsung

Prototipo	**procedure** Glcd_Init();
valori restituiti	Nessuno
Descrizione	Inizializzazione del GLCD
Vincoli	E' necessario definire preventivamente, nella parte dichiarativa del codice, i pin di controllo, le relative funzioni e la porta dati, come nell'esempio mostrato di seguito. Si noti come le linee dati appartengano tutte ad una stessa porta ad 8 bit.
Esempio	*// Interconnessioni fisiche tra PIC e GLCD* **var** GLCD_DataPort : byte **at** PORTD; **var** GLCD_CS1 : **sbit at** RB0_bit; GLCD_CS2 : **sbit at** RB1_bit; GLCD_RS : **sbit at** RB2_bit; GLCD_RW : **sbit at** RB3_bit; GLCD_EN : **sbit at** RB4_bit; GLCD_RST : **sbit at** RB5_bit; **var** GLCD_CS1_Direction : **sbit at** TRISB0_bit; GLCD_CS2_Direction : **sbit at** TRISB1_bit; GLCD_RS_Direction : **sbit at** TRISB2_bit; GLCD_RW_Direction : **sbit at** TRISB3_bit; GLCD_EN_Direction : **sbit at** TRISB4_bit; GLCD_RST_Direction : **sbit at** TRISB5_bit; *// End* Glcd_Init(); // routine di inizializzazione

Tabella 3-Routine di libreria GLCD_Init impiegata per l'inizializzazione del display (si analizzi con attenzione rispettivamente descrizione, vincoli ed esempio applicativo riportati in tabella)

Prototipo	**procedure** Glcd_Set_Side(x_pos: byte);
Valori restituiti	Nessuno
Descrizione	x_pos: posizione sull'asse x (valori: 0..127) x_pos definisce in pratica il lato del Glcd. Valori compresi tra 0 e 63 individuano il lato sinistro del display mentre valor compresi tra 64 e 127 individuano il lato destro.
Vincoli	L'impiego della routine necessita preventivamente dell'inizializzazione del GLCD (effettuata attraverso la routine *Glcd_Init*).
Esempio	Le seguenti due istruzioni sono equivalenti da un punto di vista operativo poiché entrambe selezionano il lato sinistro del display. Glcd_Set_Side(0); Glcd_Set_Side(10);

Tabella 4-Routine di libreria GLCD_Set_Side per la selezione di un banco del display grafico (si analizzi con attenzione rispettivamente descrizione, vincoli ed esempio applicativo riportati in tabella)

pagine, può essere utile selezionarne le single pagine. Per questa operazione è disponibile la funzione *Glcd_Set_Page*. Si rimanda al contenuto della Tabella 6 per maggiori dettagli.

Glcd_Read_Data (Lettura di un dato contenuto nella memoria del display)

La memoria del display può essere letta attraverso una specifica routine predefinita che prende il nome di *Glcd_Read_Data*. Ovviamente, ciò comporta la necessità di puntare preventivamente la posizione stessa attraverso le funzioni di selezione di banco, pagina e posizione precedentemente illustrate. Si rimanda al contenuto della Tabella 7 per maggiori dettagli.

Glcd_Write_Data (Scrittura di un dato nella memoria del display)

La memoria del display può essere scritta attraverso un'istruzione predefinita specifica che prende il nome di "Glcd_Write_Data".

Ovviamente, ciò comporta la necessità, come nel caso della lettura, di puntare preventivamente la posizione specifica della memoria attraverso le funzioni di selezione di banco, pagina e posizione. Si rimanda al contenuto della Tabella 8 per maggiori dettagli.

ROUTINE AVANZATE (ADVANCED ROUTINE) PER IL CONTROLLO DEL GLCD (PARTE I)

Nella gestione di un display grafico sono spesso utili routine che consentono, con una sola riga di codice ad alto livello, di realizzare effetti specifici abbastanza ricorrenti come il riempimento del display, la cancellazione di quanto visualizzato, il tracciamento di linee o rettangoli e così via. Si parla comunemente di procedure avanzate. Tra esse troviamo le seguenti:

- *Glcd_Fill* (riempimento del display);
- *Glcd_Dot* (tracciamento di un punto in una posizione specifica);
- *Glcd_Line* (tracciamento di una linea di estremi definiti);
- *Glcd_V_Line* (tracciamento di una linea verticale di posizione x definita);
- *Glcd_H_Line* (tracciamento di una linea di posizione y definita);
- *Glcd_Rectangle* (tracciamento di un rettangolo con spigoli definiti).

Le analizziamo singolarmente al fine di comprenderne appieno il corretto impiego. Premettiamo fin da ora che altre ne incontreremo nella prossima puntata del corso, puntata che potrà essere vista a tutti gli effetti come una naturale estensione della presente.

Prototipo	**procedure** Glcd_Set_X(x_pos: byte);
Valori restituiti	Nessuno
Descrizione	Posiziona il puntatore alla posizione x intesa come distanza espressa in punti dal bordo sinistro del Glcd all'interno del banco specifico. In atre parole la posizione è relativa e non assoluta perché riferita, a seconda dei casi, al banco destro o a quello sinistro. x_pos assume per questo valore compreso tra 0 e 63.
Vincoli	L'impiego della routine necessita preventivamente dell'inizializzazione del GLCD (effettuata attraverso la routine Glcd_Init).
Esempio	Glcd_Set_X(25);

Tabella 5-Routine di libreria GLCD_Set_X per la selezione della posizione all'interno di un banco del display grafico (si analizzi con attenzione rispettivamente descrizione, vincoli ed esempio applicativo riportati in tabella)

Prototipo	**procedure** Glcd_Set_Page(page: byte);
Valori restituiti	Nessuno
Descrizione	Seleziona una pagina del GLCD. Essendo disponibili 8 pagine, il parametro page può assumere valori compresi tra 0 e 7
Vincoli	L'impiego della routine necessita preventivamente dell'inizializzazione del GLCD (effettuata attraverso la routine Glcd_Init).
Esempio	Glcd_Set_Page(5);

Tabella 6-Routine di libreria GLCD_Set_Page per la selezione di una pagina del display grafico (si analizzi con attenzione descrizione, vincoli ed esempio applicativo riportati in tabella)

Prototipo	**function** Glcd_Read_Data(): byte;
Valori restituiti	Restituisce un byte dalla memoria del GLCD
Descrizione	Il byte viene letto dalla posizione corrente della memoria del GLCD. Eseguita l'operazione, il puntatore si posta alla locazione successiva.
Vincoli	L'impiego della routine necessita preventivamente di: - inizializzazione del GLCD (effettuata attraverso la routine Glcd_Init). - Scelta del banco, della posizione x e della pagina di riferimento attraverso l'impiego delle routine Glcd_Set_Side, Glcd_Set_X e Glcd_Set_Page.
Esempio	**var** data_ : byte; ... data_ := Glcd_Read_Data();

Tabella 7-Routine di libreria GLCD_Read_Data per la lettura di un dato contenuto nella memoria del display grafico (si analizzi con attenzione rispettivamente descrizione, vincoli ed esempio applicativo riportati in tabella)

Prototipo	**procedure** Glcd_Write_Data(data: byte);
Valori restituiti	Nessuno
Descrizione	Una volta fissato banco, pagina e posizione corrente del puntatore nella pagina, la funzione consente di scrivere un byte nella posizione stessa di memoria. Eseguita l'operazione, il puntatore si sposta alla posizione successiva. *Data* è il parametro (byte) il cui valore viene scritto in detta posizione di memoria.
Vincoli	L'impiego della routine necessita preventivamente di: - inizializzazione del GLCD (effettuata attraverso la routine Glcd_Init). - scelta del banco, della posizione x e della pagina di riferimento attraverso l'impiego delle routine Glcd_Set_Side, Glcd_Set_X e Glcd_Set_Page
Esempio	**var** data: byte; ... Glcd_Write_Data(data);

Tabella 8-Routine di libreria GLCD_Write_Data per la scrittura di un dato nella memoria del display grafico (si analizzi con attenzione rispettivamente descrizione, vincoli ed esempio applicativo riportati in tabella)

Prototipo	**procedure** Glcd_Fill(pattern: byte);
Valori restituiti	Nessuno
Descrizione	Riempie l'intero display di uno dei due colori disponibili. Il parametro pattern può assumere valore 0 per pulire completamente il display oppure valore 0XFF per colorarlo completamente.
Vincoli/operazioni preliminari	Il display necessita di essere preventivamente inizializzato attraverso la routine Glcd_Init.
Esempio	*// pulisce il display* Glcd_Fill(0); // riempie il display Glcd_Fill(0xFF).

Tabella 9-Routine di libreria GLCD_Fill per il riempimento del display grafico (si analizzi con attenzione rispettivamente descrizione, vincoli ed esempio applicativo riportati in tabella)

Prototipo	**procedure** Glcd_Dot(x_pos: byte; y_pos: byte; color: byte);
Valori restituiti	Nessuno
Descrizione	Traccia un punto sul GLCD nella posizione di coordinate (x_pos, y_pos). Significato dei Parametri: - x_pos: posizione x (Valore compreso tra 0 e 127); - y_pos: posizione y (Valore compreso tra 0..63); - color: colore (Valore compreso tra 0 e 2 (0→cancella il punto, 1→traccia il punto; 2→inverte lo stato del punto).
Vincoli/operazioni preliminari	L'impiego della routine necessita preventivamente dell'inizializzazione del GLCD (effettuata attraverso la routine Glcd_Init).
Esempio	*// Inverte lo stato del punto della posizione (0,0) estrema sinistra in alto* Glcd_Dot(0, 0, 2);

Tabella 10-Routine di libreria GLCD_Dot per il tracciamento di un punto sul display grafico (si analizzi con attenzione rispettivamente descrizione, vincoli ed esempio applicativo riportati in tabella)

Glcd_Fill (Riempimento del display)

Il riempimento dell'intero display è un'operazione che può risultare utile in diverse occasioni, specie quando si vogliono realizzare effetti grafici di comparsa e scomparsa. Tale operazione è realizzabile attraverso l'impiego della procedura *Glcd_Fill*. Si rimanda al contenuto della Tabella 9 per maggiori dettagli.

Glcd_Dot (Rappresentazione di un punto)

Il punto è l'elemento base di qualunque entità grafica aspetto da cui scaturisce la disponibilità di una routine che consente di rappresentare dei punti sul GLCD. Si tratta della routine Glcd_Dot. Ovviamente, la routine necessita del passaggio delle coordinate del punto. Si rimanda al contenuto della Tabella 10 per maggiori dettagli.

Glcd_Line (Tracciamento di una linea)

La linea è un altro elemento grafico importante e ricorrente nell'impiego di un display grafico. Il suo tracciamento è reso possibile dalla procedura di libreria *Glcd_Line*. Essendo una linea o meglio un segmento definito dai due punti estremi, il tracciamento di questa entità grafica necessita del passaggio alla routine delle coordinate di questi punti. Si rimanda al contenuto della Tabella 11 per maggiori dettagli.

Prototipo	**procedure** Glcd_Line(x_start: integer; y_start: integer; x_end: integer; y_end: integer; color: byte);
valori restituiti	nessuno
Descrizione	Disegna una linea sul GLCD Significato dei parametri: • x_start: punto di partenza, coordinata x. Valori validi: 0..127 • y_start: punto di partenza, coordinate y. Valori validi: 0..63 • x_end: punto finale, coordinate x. Valori validi: 0..127 • y_end: punto finale coordinate y. Valori validi: 0..63 • color: parametro colore. Valori validi: 0..2(0→cancella punto, 1→ traccia punto, 2→ inverte lo stato del punto.
Vincoli/operazioni preliminari	L'impiego della routine necessita preventivamente dell'inizializzazione del GLCD (effettuata attraverso la routine Glcd_Init).
Esempio	*// traccia una linea avente punti estremi (0,0) e (20,30)* Glcd_Line(0, 0, 20, 30, 1);

Tabella 11-Routine di libreria GLCD_Line per il tracciamento di una linea sul display grafico (si analizzi con attenzione rispettivamente descrizione, vincoli ed esempio applicativo riportati in tabella)

Glcd_V_Line (Tracciamento di una linea verticale)

Per il tracciamento di una linea verticale è disponibile la routine *Glcd_V_Line*. Trattandosi di una linea verticale è sufficiente passare alla routine tre soli parametri: coordinata y iniziale, coordinata y finale e coordinata x. La coordinata x del punto inziale e finale infatti coincidono. Si rimanda al contenuto della Tabella 12 per maggiori dettagli. Ciò non toglie ovviamente, la possibilità di utilizzare la funzione *Glcd_Line* anche per il tracciamento di linee verticali.

Glcd_H_Line (Tracciamento di una line orizzontale)

Analogamente a quanto visto per il tracciamento di linee verticali, per il tracciamento di una linea orizzontale è resa disponibile la routine *Glcd_H_Line*. Trattandosi di una linea orizzontale è sufficiente passare tre soli parametri: coordinata x iniziale, coordinata x finale e coordinata y. La coordinata y del punto inziale e finale infatti coincidono. Si rimanda al contenuto della Tabella 13 per maggiori dettagli. Ciò non toglie ovviamente, la possibilità di utilizzare la funzione *Glcd_Line* anche per il tracciamento di linee orizzontali.

Glcd_Rectangle (Tracciamento di un rettangolo)

Il rettangolo è un'entità che viene rappresentata abbastanza ricorrentemente nella gestione di un display LCD grafico. La routine *Glcd_Rectangle* consente di eseguire il tracciamento di un rettangolo. I parametri che vengono passati alla routine sono le coordinate (x,y) del punto in alto a sinistra e quelle del punto in basso a destra. Si

Prototipo	**procedure** Glcd_V_Line(y_start: byte; y_end: byte; x_pos: byte; color: byte);
valori restituiti	Nessuno
Descrizione	Traccia una linea verticale sul display. Significato dei parametri: - y_start: coordinata y del punto di inizio. Valori validi: 0..63; - y_end: coordinata y del punto finale. Valori validi: 0..63; - x_pos: coordinate x della linea verticale. Valori validi: 0..127; - color: parametro colore. Valori validi: 0..2 (0→cancella punto, 1→ traccia punto, 2→inverti stato del punto).
Vincoli/operazioni preliminari	L'impiego della routine necessita preventivamente dell'inizializzazione del GLCD (effettuata attraverso la routine Glcd_Init).
Esempio	*// Disegna una linea verticale tra I punti (10,5) e (10,25)* Glcd_V_Line(5, 25, 10, 1);

Tabella 12-Routine di libreria GLCD_V_Line per il tracciamento di una linea verticale sul display grafico (si analizzi con attenzione rispettivamente descrizione, vincoli ed esempio applicativo riportati in tabella)

Prototipo	**procedure** Glcd_V_Line(x_start: byte; x_end: byte; y_pos: byte; color: byte);
Valori restituiti	Nessuno
Descrizione	Traccia un alinea orizzontale sul display. Significato dei parametri: - x_start: coordinate x del punto di inizio. Valori validi: 0..127; - x_end: coordinate x del punto di fine. Valori validi: 0..127; - y_pos: coordinate y della linea orizzontale. Valori validi: 0..63; - color: parametro colore. Valori validi: 0..2 (0→cancella punto, 1→ traccia punto, 2→inverti stato del punto.
Vincoli/operazioni preliminari	L'impiego della routine necessita preventivamente dell'inizializzazione del GLCD (effettuata attraverso la routine Glcd_Init).
Esempio	*// Traccia una linea orizzontale tra i punti (10,20) e (50,20)* Glcd_H_Line (10, 50, 20, 1);

Tabella 13-Routine di libreria GLCD_H_Line per il tracciamento di una linea orizzontale sul display grafico (si analizzi con attenzione rispettivamente descrizione, vincoli ed esempio applicativo riportati in tabella)

rimanda al contenuto della Tabella 14 per maggiori dettagli.

ESEMPI FIRMWARE

In questa seconda parte dell'articolo vogliamo presentare un esempio dimostrativo di firmware critto in MikroPascal che impiega le principali routine di libreria, presentate fino a questo momento, per la gestione del GLCD Samsung KS0108. Il codice (Figura 6), come già accennato, non presenta un vero e proprio scopo applicativo ma piuttosto dimostrativo. Il firmware "*.HEX*" generato dalla compilazione del codice MikroPascal può essere testato con la scheda di sviluppo EasyPIC3 o con schede di sviluppo successive della stessa serie o ancora con simulatori software come Real PIC Simulator. Lo schema elettrico di riferimento è per questo quello di

Prototipo	**procedure** Glcd_Rectangle(x_upper_left: byte; y_upper_left: byte; x_bottom_right: byte; y_bottom_right: byte; color: byte);
valori restituiti	Nessuno
Descrizione	Traccia un rettangolo sul display. Significato dei parametri: - x_upper_left: coordinata x dell'angolo in alto a sinistro. Valori validi: 0..127; - y_upper_left: coordinata y dell'angolo in alto a sinistra. Valori validi: 0..63; - x_bottom_right: coordinata x dell'angolo in basso a destra. Valori validi: 0..127; - y_bottom_right: coordinata y dell'angolo in basso a destra. Valori validi: 0..63; - color: parametro colore. Valori validi: 0..2 (0→cancella punto, 1→ traccia punto, 2→inverti stato del punto.
Vincoli/operazioni preliminari	L'impiego della routine necessita dell'inizializzazione del GLCD (effettuata attraverso la routine Glcd_Init).
Esempio	*// traccia un rettangolo tra I punti (5,5) e (40,40)* Glcd_Rectangle(5, 5, 40, 40, 1);

Tabella 14-Routine di libreria GLCD_Rectangle per il tracciamento di un rettangolo sul display grafico (si analizzi con attenzione rispettivamente descrizione, vincoli ed esempio applicativo riportati in tabella)

Figura 5. E' interessante notare le interconnessioni tra PIC e GLCD a conferma di quanto detto precedentemente sul significato delle linee di controllo e di quelle dati. In ogni caso, non è difficile modificare, nel firmware (codice MikroPascal), la corrispondenza tra pin del controllore e pin del display al fine di adattare la configurazione di questi ultimi all'hardware di altre schede di sviluppo.

A tale scopo, volendo ancora utilizzare l'esempio applicativo di Figura 6 sarà sufficiente modificare la sezione "Definizione delle interconnessioni PIC-Glcd" in maniera tale da adattarla alle specifiche esigenze. Il lettore potrà modificare il codice facilmente osservandone gli effetti sul GLCD.

Il programmatore, per gestire il GLCD, si limita semplicemente a richiamare tali routine passando ad esse i valori necessari. In questo modo, con una sola istruzione di alto livello, è possibile rappresentare box, linee, circonferenze, aree piene o vuote, e dare origine a diversi effetti grafici più o meno elaborati. Analogamente, come avremo modo di osservare nella prossima puntata è possibile gestire la rappresentazione di scritte e immagini.

Il codice di esempio MikroPascal è riportato in Figura 6. Si riconoscono, perché indicate, rispettivamente le seguenti sezioni:

- *Definizione delle interconnessioni PIC-Glcd;*
- *Sezione MAIN.*

All'interno della sezione main si riconoscono a loro volta e nell'ordine le seguenti operazioni implementate attraverso le routine di libreria precedentemente illustrate:

- *Inizializzazione GLCD;*
- *riempimento GLCD;*
- *pulizia GLCD;*
- *tracciamento di punti;*
- *tracciamento di linee oblique;*
- *tracciamento linee verticali;*
- *tracciamento linee orizzontali;*
- *tracciamento di rettangoli;*

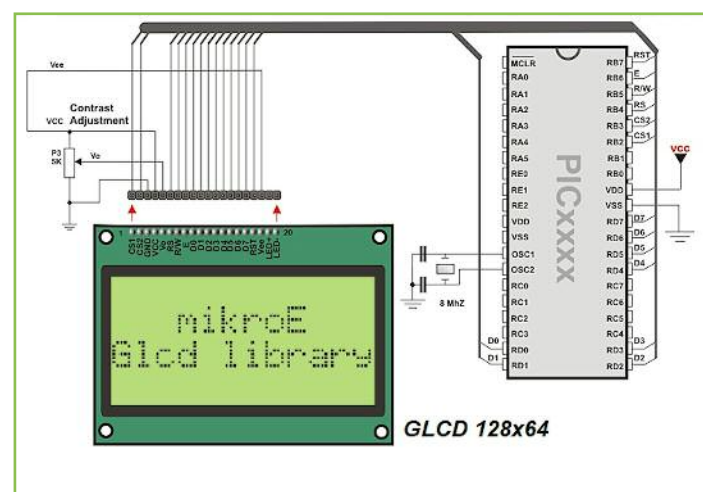



Figura 5-Schema elettrico di interfacciamento PIC-GLCD 128x64 KS0108. Lo schema è lo stesso utilizzato per l'interconnessione PIC-GLCD integrata sulle schede di sviluppo EasyPIC di Mikroelektronika [1], [2].

Nella stessa Figura 6 è riportato lo stato del GLCD in diversi momenti dell'esecuzione a conferma delle operazioni rese possibili dalle singole routine.
Dopo aver realizzato il circuito, secondo lo schema di Figura 5, eseguita la stesura del firmware, compilato il sorgente e programmato il PIC con il file .HEX ottenuto è possibile verificare la rispondenza del comportamento del GLCD a quanto atteso. Se non è visualizzato nulla sul display, ma si è certi che il circuito sia stato implementato correttamente dal punto di vista elettrico e che il PIC sia ben programmato è opportuno controllare la regolazione del contrasto agendo sull'apposito trimmer di regolazione (pin Vo). Come evidenzia la Figura 5 è stato impiegato il GLCD128x64 pixel KS0108. In Figura 6 è mostrato sia il listato MikroPascal del codice che l'effetto dell'esecuzione dei singoli cicli interni. L'esecuzione mostra come sia semplice riprodurre effetti grafici di natura geometrica sul display grafico mediante l'impiego di routine di libreria predefinite. Ovviamente, ciò non toglie la possibilità al programmatore di utilizzare routine predefinite di libreria per comporne di proprie.

CONCLUSIONI

Quanto riportato in questa lezione è sufficiente per realizzare applicazioni firmware con controllori PIC scritte in codice MikroPascal e che prevedono l'impiego di display grafici.
Le routine avanzate presentate nel precedente paragrafo non sono le uniche messe a disposizione da MikroPascal PRO for PIC. Nella prossima lezione ne presenteremo e analizzeremo altre altrettanto interessanti e soprattutto semplici da utilizzare. Tra queste vi sono le routine che consentono di gestire la riproduzione sul display grafico di vere e proprie immagini bitmap, ovviamente adeguatamente formattate. A tale proposito risulterà di grande utilità un potente tool integrato in MikroPascal PRO for PIC: il *GLCD Bitmap Editor* che analizzeremo appositamente ed utilizzeremo negli esempi firmware della prossima lezione. La prossima lezione rappresenterà, a tutti gli effetti, la naturale continuazione e il naturale completamento dell'argomento introdotto con questa puntata del corso.

Per approfondire

[1] PIC16F87X Data Sheet 28/40-Pin 8-Bit CMOS FLASH Microcontrollers (www.mikrochip.com)
[2] Help in linea MikroPascal Pro for PIC v. 5.61
[3] MikroPascal PRO for PIC v.5.61 demo e documentazione relativa (www.mikroe.com)
[4] Development Tools Easy PIC 3-Board Schematic with functional schematic.

```
{*
 * Corso: MikroPascal PRO for PIC-Fare Elettronica-Lezione n.: 10
 * Nome progetto: GLCD_test-Rev.0: 2013-05-26; Autore: A. Giannico
 * Descrizione: Il codice mostra il funzionamento di alcune routine di libreria
                per il controllo dei display grafici (GLCD). In particolare
                viene utilizzato un display 128x64 standard KS0108.
                L'assegnazione dei pin dele porte del PIC è fatta in maniera da
                rendere il codice compatibile con EasyPIC3.
 * MCU:PIC18F877; Sch. di sviluppo:Easypic3; Oscillatore: HS 8.0000 MHz
 * SW: mikroPascal PRO for PIC  ver. 5.61
   http://www.mikroe.com/eng/products/view/10/mikropascal-pro-for-pic/
 *}
//____________________________________________________________________________
program GLCD_Test;
var counter, dato, i,j,k : byte;
    someText : array[18] of char;
//_____________________________________Definizione delle interconnessioni PIC-Glcd
var GLCD_DataPort : byte at PORTD;  // port DATI
                                    // PIC <-----> GLCD
                                    //__________________
                                    // RD0 <-----> D0
                                    // RD1 <-----> D1
                                    // .........
                                    // RD7 <-----> D7
                                    //__________________
                                    // pin di controllo
var GLCD_CS1 : sbit at RB2_bit;
    GLCD_CS2 : sbit at RB3_bit;
    GLCD_RS  : sbit at RB4_bit;
    GLCD_RW  : sbit at RB5_bit;
    GLCD_EN  : sbit at RB6_bit;
    GLCD_RST : sbit at RB7_bit;

var GLCD_CS1_Direction : sbit at TRISB2_bit;
    GLCD_CS2_Direction : sbit at TRISB3_bit;
    GLCD_RS_Direction  : sbit at TRISB4_bit;
    GLCD_RW_Direction  : sbit at TRISB5_bit;
    GLCD_EN_Direction  : sbit at TRISB6_bit;
    GLCD_RST_Direction : sbit at TRISB7_bit;
// ___________________________________________________________________________
// _______________________________________________________________MAIN________
begin
  Glcd_Init();                                     // Inizializzazione GLCD
  while TRUE do
          begin
          Glcd_Fill(0xFF);                         // riempimento GLCD
          Delay_ms(300);
          Glcd_Fill(0x00);                         // pulizia GLCD
          Delay_ms(300);
          Glcd_Fill(0xFF);                         // Glcd_Fill --> riempimnto GLCD
          for k:=0 to 2  do                        // tracciamento di punti
                begin
                for i:=0 to 127  do
                      begin
                      for j:=0 to 63 do
                           begin
                           Delay_ms(100);
                           Glcd_Dot(i, j, k); // punto (i,j)
                           end;            // Glcd_Dot --> rappresentazione
                      end;                 //               di un punto
                end;
          Delay_ms(300);
          Glcd_Fill(0xFF);                         // riempimento GLCD
          //_________________________________________________________________
          for k:=0 to 2  do                        // tracciamento di linee oblique
                begin
                for i:=0 to 127  do
                      begin
                      for j:=0 to 63 do
                           begin
                           Delay_ms(100);
                           Glcd_Line(0, 0, i, 32, k);//linea (00)-(i,32)
                           end;            // Glcd_Line --> rappresentazione
                      end;                 //               di una linea
                end;
                //___________________________________________________________
                for k:=0 to 2  do          // tracciamento linee verticali
                      begin                // da y=10 a y=50 x=j
                      for j:=0 to 63 do
                           begin
                           Delay_ms(100);
                           Glcd_V_Line(10, 50, j, k);
                           end;            // Glcd_V_Line --> rappresentaz.
                      end;                 // di una linea verticale
                //___________________________________________________________
                for k:=0 to 2  do          // tracciamento linee orizzontali
                      begin                // da x=10 ad x=120 y=j
                      for j:=10 to 50 do
                          begin
                          Delay_ms(100);
                          Glcd_H_Line(10, 120, j, k);
                          end;             // Glcd_H_Line --> rappresentaz.
                      end;                 //  di un alinea orizzontale
              //_____________________________________________________________
                for k:=0 to 2  do          // tracciamento di rettangoli
                      begin
                      for i:=0 to 32  do
                          begin
                          Delay_ms(100);
                          Glcd_Rectangle(63-i, 32-i, 63+i, 32+i, k);
                          end;             // Glcd_Rectangle --> rappresent.
                      end;                 // di un rettangolo
                Glcd_Fill(0x00);           // pulisci GLCD
    end;
end.
```

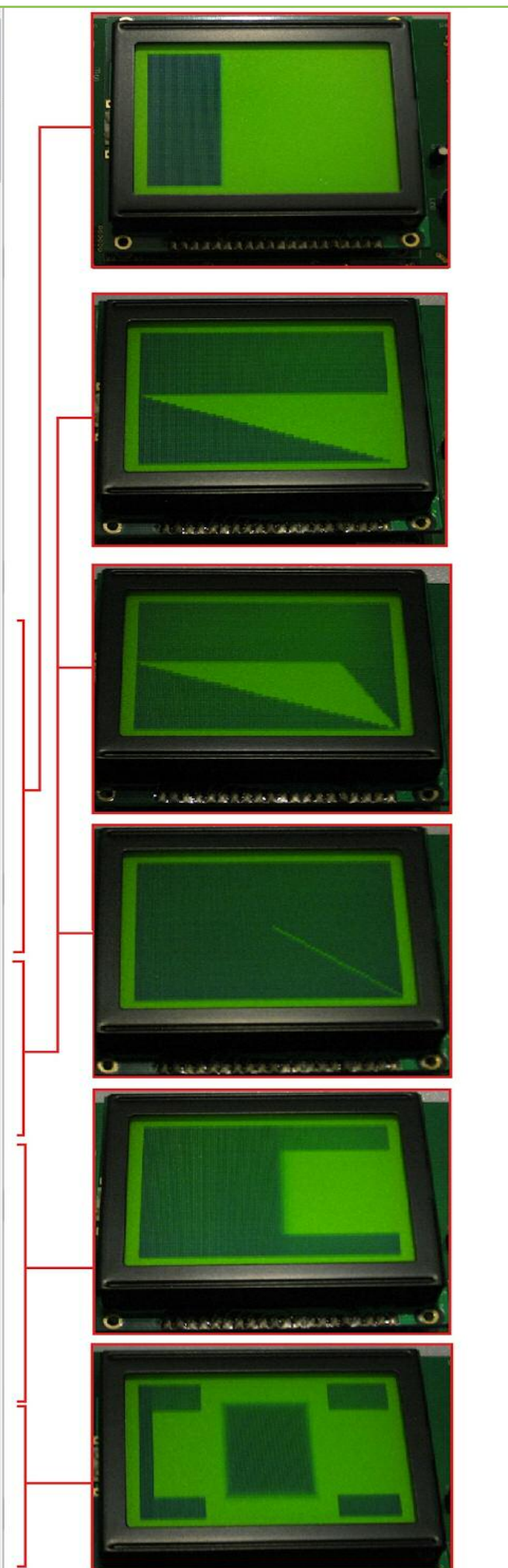



Figura 6-Codice di esempio Glcd_Test ed effetti prodotti sul display grafico KS0108.

di William Pasin

PROGETTARE CON L'USB

L'USB è oramai diventato uno standard di comunicazione ed ha completamente reso obsoleto il classico bus seriale RS232. Vediamo allora come integrare l'usb nei propri progetti utilizzando i PIC con USB a bordo

A differenza di un normale bus seriale RS232, la trasmissione dati tramite l'Universal Serial Bus può essere abbastanza difficoltosa per il progettista che voglia sviluppare da zero l'intero protocollo. Le specifiche USB 2.0 sono, infatti, un documento di oltre 650 pagine. Fortunatamente molti produttori di microcontrollori hanno cercato di facilitare il lavoro del progettista, realizzando dispositivi che gestiscono interamente il protocollo USB, rendendolo molto semplice ed intuitivo nell'uso, come fosse una semplice interfaccia seriale.

Nel presente articolo si analizzeranno in dettaglio i microcontrollori Microchip PIC18F2455/2550/4455/4550. Rispetto agli altri modelli della famiglia 18F, questi micro sono dotati di controller USB compatibile con lo standard 2.0. Supportano le velocità Low Speed (1.5Mbits/s) e Full Speed (12Mbits/s); implementano tutti i tipi di trasferimento previsti dalla specifica (Control, Interrupt, Isocrono e Bulk) e supportano fino a 32 endpoint, di cui 16 bidirezionali.

VELOCITÀ DEL PROTOCOLLO USB			
Standard	Nome	Valore	Tolleranza sul segnale
USB 1.1	Low Speed	1.5Mbits/s	±15000ppm
	Full Speed	12Mbits/s	±2500ppm
USB 2.0	High Speed	480Mbits/s	±500ppm

Tabella 1

Si cercherà ora di focalizzare l'attenzione su alcuni concetti base del protocollo USB, per essere poi pronti ad affrontare un esempio applicativo con i PIC sopra descritti. In particolare, si mostrerà come emulare una porta seriale utilizzando una connessione USB di un PC. Questo è un problema che si presenta spesso con i moderni computer che ormai non sono dotati di porta seriale, mentre molte applicazioni embedded comunicano con il PC proprio con tale interfaccia. L'emulazione RS-232 è un buon compromesso, poiché non richiede la modifica dell'applicazione software implementata sul PC; quest'ultima, infatti, continuerà a credere di utilizzare un COM anziché una porta USB.

USB OVERVIEW

Secondo lo standard 1.1, l'USB supporta due differenti velocità di trasmissione: Full Speed a 12Mbits/s e Low Speed a 1.5Mbits/s. La soluzione più lenta è meno suscettibile alle interferenze elettromagnetiche (EMI) e quindi più economica poiché si può risparmiare sulla qualità degli ele-

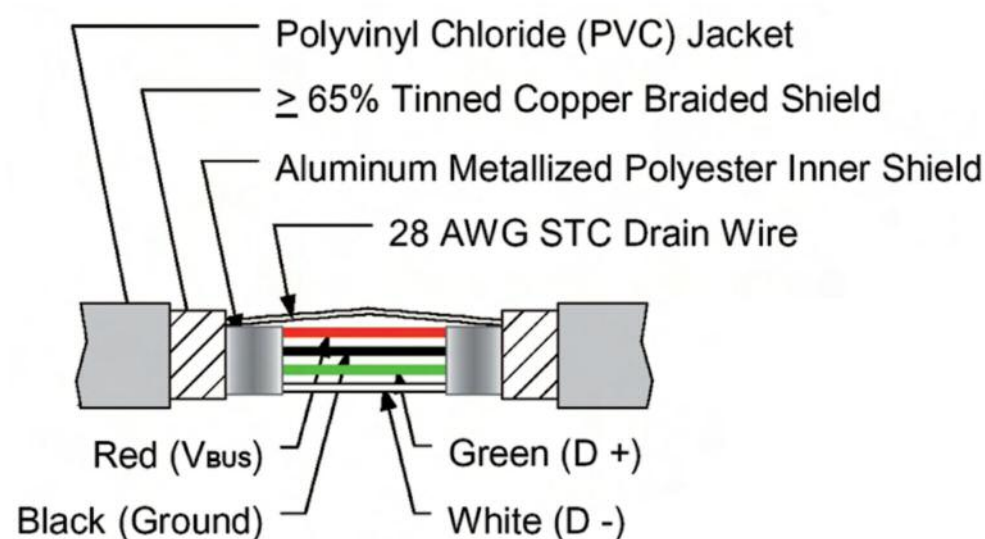



Figura 1 Schema del cavo USB

menti impiegati. L'USB 2.0 ha aumentato di 40 volte la massima velocità della versione 1.1. I 480Mbits/s sono anche indicati con il termine High Speed e rappresentano un manovra commerciale per competere con la tecnologia concorrente, la Firewire 1394, che raggiunge invece la velocità massima di 400Mbits/s (almeno per quanto riguarda lo standard FireWire 400). La tabella 1 riassume quanto sopra esposto. Il bus USB è di tipo host controlled. Ci può essere soltanto un host per ogni bus. La specifica non prevede nessuna forma di sistema multimaster, anche se con l'USB OTG (On-The-Go) sembrerebbe che tale regola sia violata. Il protocollo HNP (Host Negotiation Protocol), infatti, consente a due dispositivi di negoziare per il ruolo di host; in ogni caso, al termine di tale processo, il dispositivo host è sempre uno e uno solo. L'host si occupa di stabilire tutte le transazioni e di ripartire la banda tra i vari dispositivi. I dati possono essere inviati mediante differenti metodi, usando un protocollo token-based. Come il nome stesso suggerisce, l'USB è un bus seriale. Esso utilizza 4 cavi schermati di cui 2 per l'alimentazione (+5V e GND). I rimanenti due costituiscono una coppia intrecciata per ottenere un segnale differenziale. La Figura 1 riporta un'immagine del cavo, dei colori standard e della schermatura, mentre la tabella 2 elenca le corrispondenze cavi/colore. Lo schema di codifica utilizzato è la NRZI (Non Return to Zero Invert) e per la sincronizzazione dei clock del ricevitore e del trasmettitore si usa un segnale di sincronismo in testa al pacchetto trasmesso. Una delle caratteristiche di successo dell'USB è di essere plug&play, con caricamento dinamico dei driver. L'host rileva l'inserimento del dispositivo USB e lo interroga per sapere quale driver caricare. La capacità dell'host di decidere il driver corretto è possibile grazie alla combinazione di PID (Product ID) e VID (Vendor ID). Il VID è fornito dall'*USB Implementor's forum* dietro compenso. Altre organizzazioni forniscono un extra VID per attività non commerciali come insegnamento, ricerca o per hobbisti in forma gratuita. Questa è anche la soluzione adottata da Microchip che fornisce una serie di VID/PID utilizzabili.

PIN	COLORE DEL CAVO	FUNZIONE
1	Rosso	VBUS (+5V)
2	Bianco	D-
3	Verde	D+
4	Nero	Ground

Tabella 2

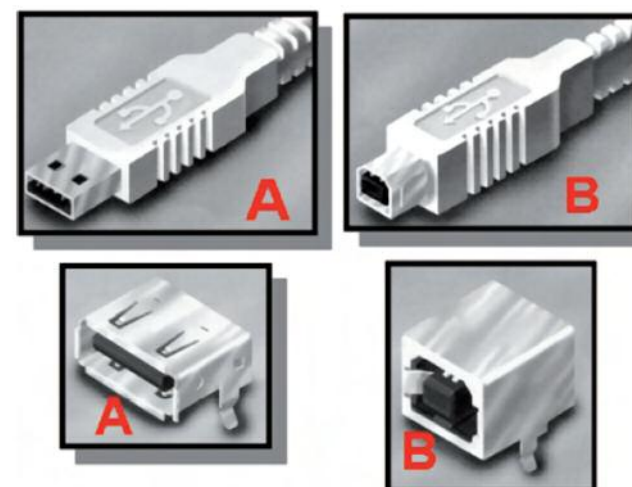



Figura 2 Connettori USB di tipo A e B

CONNETTORI

Tutti i dispositivi hanno una connessione di upstream per dialogare con l'host e tutti gli host hanno una connessione di downstream per interfacciarsi con i dispositivi. Ovviamente, onde evitare errori di connessione, i due connettori (upstream e downstream) non sono intercambiabili. La specifica USB 1.1 introduce due tipi di connettori (in figura 2 sono riportati sia le prese che i plug):

1. Tipo A
2. Tipo B

Figura 3 Connettori mini B

Con lo standard 2.0 si è introdotto il connettore mini B (figura 3). Il motivo di questo cambiamento è da imputare soprattutto alla necessità di integrazione dei dispositivi elettronici (come cellulari o lettori MP3). Infine, la specifica relativa a USB OTG ha introdotto ulteriori connettori: mini-A e mini AB.

CARATTERISTICHE ELETTRICHE

USB utilizza un coppia differenziale per la trasmissione dei dati (D- e D+). Nel caso di USB 1.1, un livello logico 1 è trasmesso quando D+ è portato alla tensione di 2.8V mediante un resistore di pull-down il cui valore è 15ko e D- ad una tensione di 0.3V con un resistore di pull-up il cui valore è 1.5k. Il ricevitore capirà di ricevere 1 se D+ è superiore rispetto a D- di 200mV, mentre interpreterà 0 se D+ è inferiore di 200mV rispetto a D.
Un transceiver USB deve essere in grado di generare oltre che segnali differenziali anche segnali single ended, in corrispondenza di certi stati del bus. Ad esempio, uno stato logico 0 di tipo single ended (SE0) può essere utilizzato per indicare il reset del dispositivo se tenuto per più di 10ms. Lo stato SE0 è ottenuto mantenen-

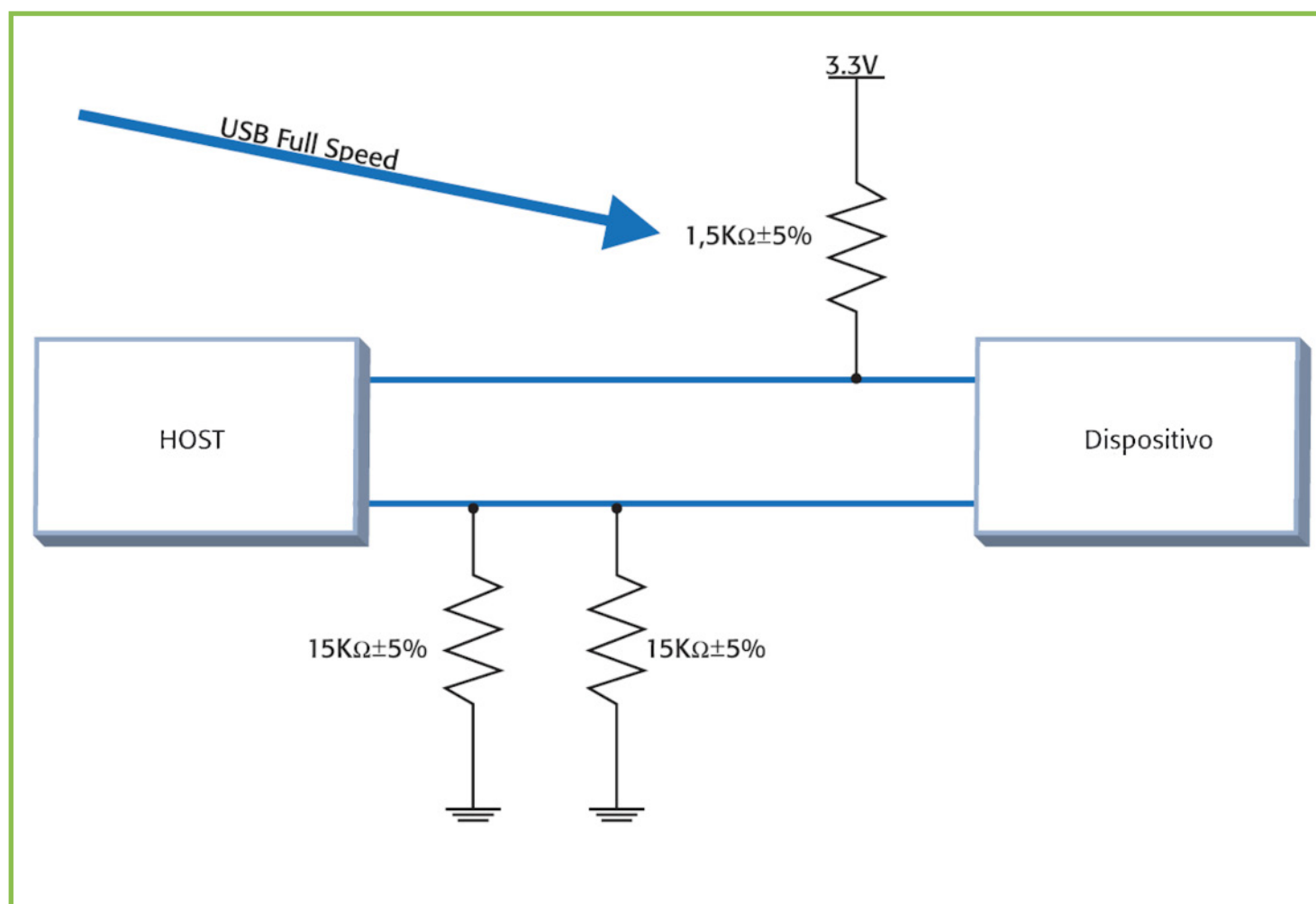



Figura 4 Identificazione dispositivo Full Speed

Figura 5 Identificazione dispositivo Low Speed

do sia D- che D+ al livello basso (<0.3V). Come ricordato in precedenza i PIC18F2455/2550/4455/4550 implementano internamente un transceiver che rispetta tali specifiche. L'indicazione della velocità del dispositivo è effettuata alla connessione, portando al livello alto entrambe le linee dati.

Un dispositivo Full Speed utilizza un resistore di pull-up sulla linea D+, come mostrato in figura 4.Tale resistore viene anche usato dall'host per rilevare la presenza di un dispositivo connesso alla sua porta. Senza tale resistenza l'host considera che nessun dispositivo è collegato. Spesso tale componente è realizzato on-chip sul dispositivo, mentre in altri casi è necessario aggiungerlo esternamente.

Nel caso dei PIC è possibile scegliere se utilizzare i resistori interni oppure quelli esterni. I chip sono inoltre dotati di un regolatore con uscita a 3.3V, che consente di alimentare il chip con una tensione singola a 5V. Nel caso di dispositivo Low Speed si utilizza la stessa tecnica, ma con il resistore sulla linea D-, come illustrato in figura 5. Il discorso è leggermente diverso per i dispositivi High Speed.

Inizialmente la connessione viene eseguita come un Full Speed (1.5Ko a 3.3V). In seguito, durante la fase di reset, il dispositivo invierà all'host una sequenza per segnalare la velocità di 480Mbits/s.

Un'altra caratteristica importante dello standard USB è la possibilità di trasportare l'alimentazione sul bus. Da questo punto di vista si distinguono due differenti classi di dispositivi:

1. Bus-power. Prelevano l'alimentazione dal bus.

2. Self-power. Prelevano l'alimentazione da una sorgente esterna.

Ovviamente, per la prima categoria è specificato un limite alla potenza assorbita dal bus. A ciascun dispositivo sono assicurati 100mA@5V (quindi una potenza di 500mW). Una potenza addizionale può essere richiesta fino ad un massimo di 500mA. La specifica USB definisce anche una modalità di sospensione (*Suspend Mode*). In tale condizione la corrente massima non deve superare i 500mA. Un dispositivo entra in suspend mode se rimane inattivo per oltre 3ms.

ENUMERAZIONE E PROTOCOLLO

Quando il dispositivo è collegato al bus, l'host avvia un processo di enumerazione allo scopo di identificarlo. In pratica, l'host interroga il dispositivo, raccoglie informazioni utili come consumo di potenza, velocità e protocollo utilizzato. La figura 7 riporta le varie fasi che costituiscono tale processo. Il protocollo USB è basato su particolari pacchetti costituiti dai seguenti campi (i vari tipi di pacchetti sono riportati in Figura 8):

- **Sync**. Tutti i pacchetti devono iniziare con questo campo. È formato da 8bit e serve per la sincronizzazione del ricevitore con il trasmettitore.
- **PID (Packet IDentification)**. Serve per identificare il tipo di pacchetto che si de-

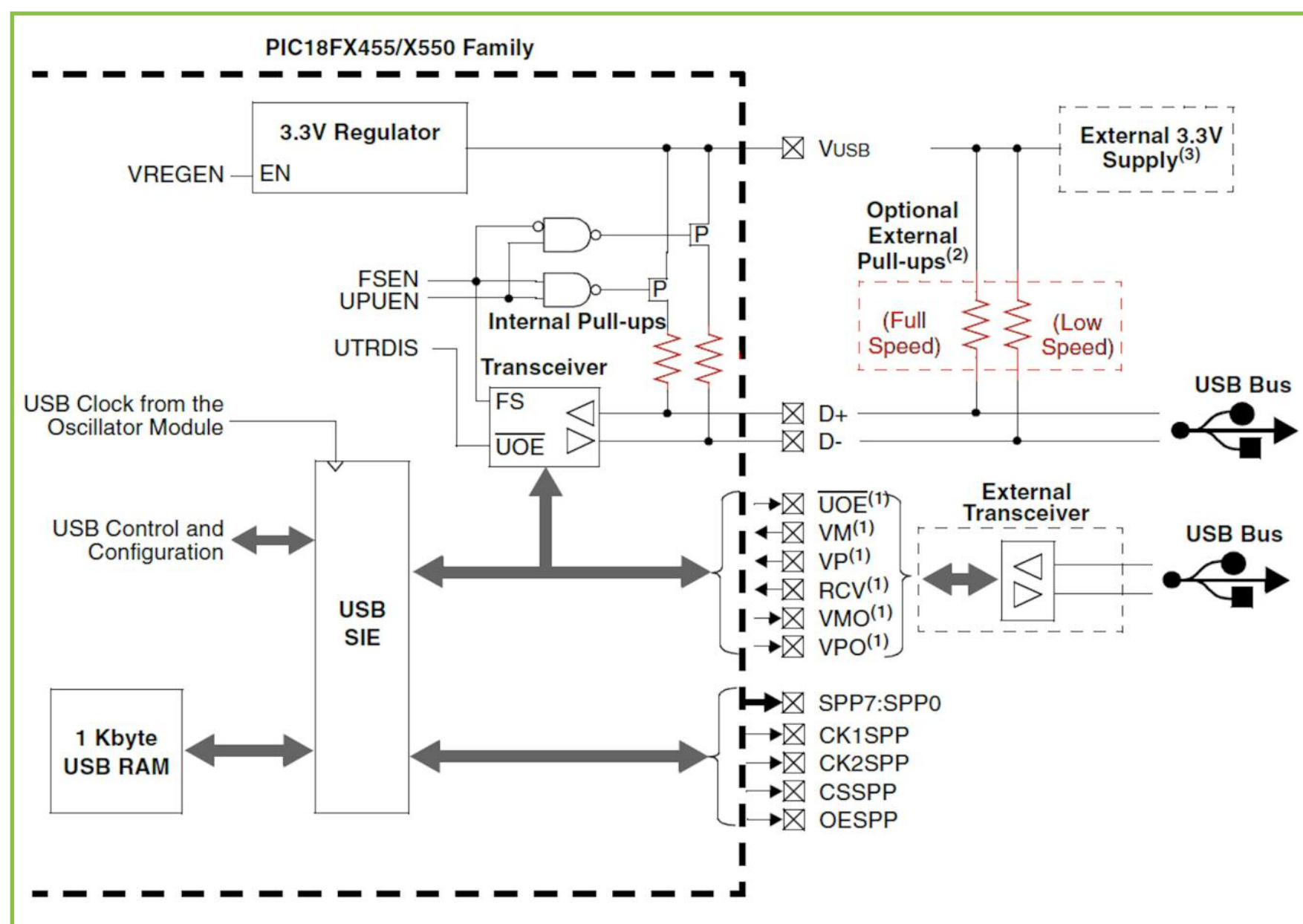



Figura 6 Resistenze di pull-up interne ed esterne nei PIC

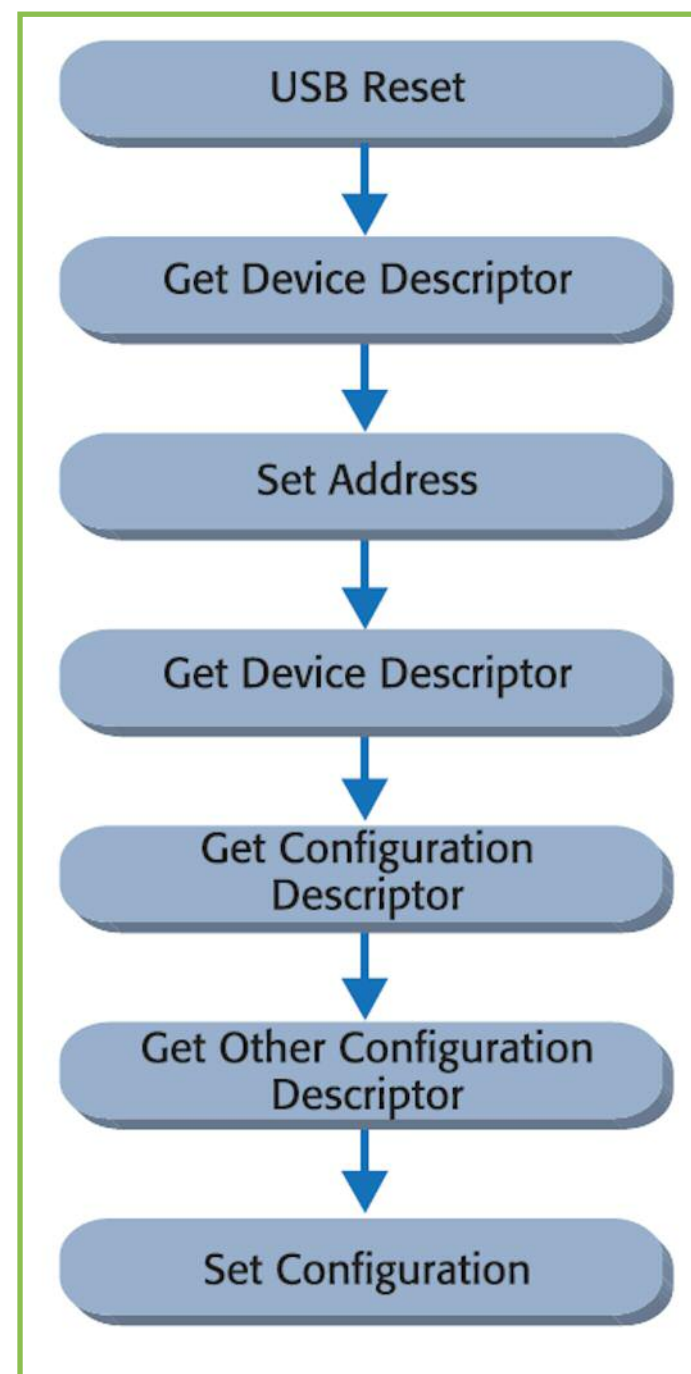



Figura 7 Diagramma di flusso del processo di enumerazione

GRUPPO	VALORE PID	TIPO PACCHETTO
Token	0001	OUT Token
	1001	IN Token
	0101	SOF Token
	1101	SETUP Token
Data	0011	DATA0
	1011	DATA1
	0111	DATA2
	1111	MDATA
Handshake	0010	ACK Handshake
	1010	NAK Handshake
	1110	STALL Handshake
	0110	NYET (No Response Yet)
Special	1100	PREamble
	1100	ERR
	1000	Split
	0100	Ping

Tabella 3

ve trasmettere. La tabella 3 elenca tutte le possibili combinazioni di questo campo.

- **ADDR**. Specifica a quale dispositivo è destinato il pacchetto. È costituito da 7bit e quindi 27 = 127 dispositivi complessivamente possono essere presenti su di un bus USB.
- **ENDP**. Questo campo è formato da 4bit e quindi 16 possibili endpoints. I dispositivi low speed sono però dotati solo di massimo 4 endpoints. Gli endpoints possono essere visti come sorgenti/destinazione di dati. Essi sono una sorta di interfaccia tra l'hardware ed il firmware del dispositivo (figura 9).
- **CRC**. Serve per garantire l'integrità dei dati.
- **EOP**. Indica la fine del pacchetto.

Un altro aspetto interessante del protocollo sono le modalità di trasmissione. La specifica ne riporta 4 complessivamente:

- **Isocrona**. Utile soprattutto per trasmettere grandi quantità di dati (video e audio), senza però nessuna garanzia sull'integrità dei dati.
- **Bulk**. Simile alla prima, ma garantisce l'integrità.
- **Interrupt**. È destinata alla trasmissione di ridotte quantità di dati, in cui però è essenziale garantire che i dati non siano corrotti.
- **Control**. Utilizzata per il setup del dispositivo. È importante sottolineare che i dispositivi low speed sono dotati solo delle ultime due modalità di trasferimento.

UTILIZZARE I PIC18FX455/X550 PER APPLICAZIONI USB

La figura 10 riporta lo schema completo dell'architettura implementata a bordo del

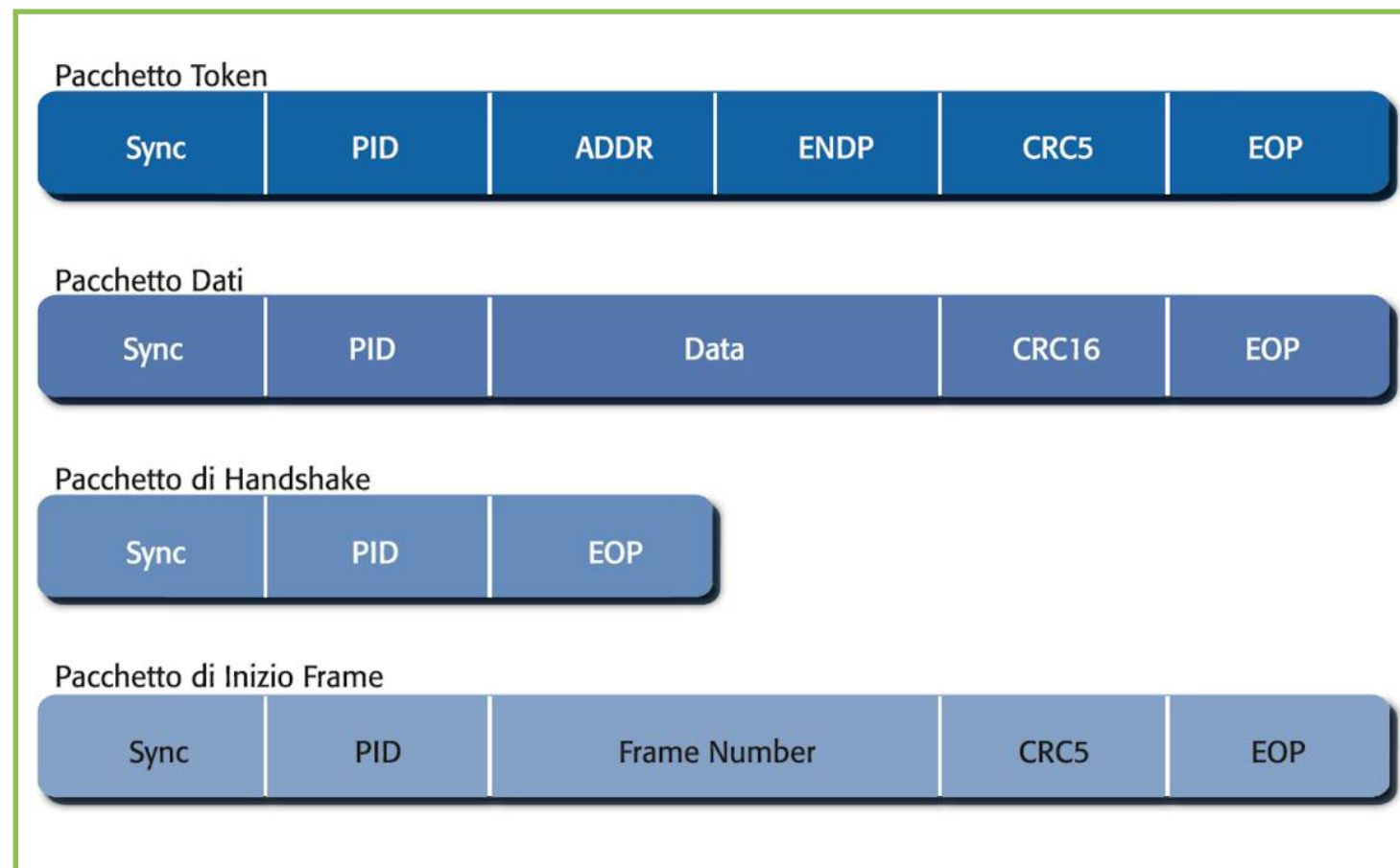



Figura 8 Formato del pacchetto USB

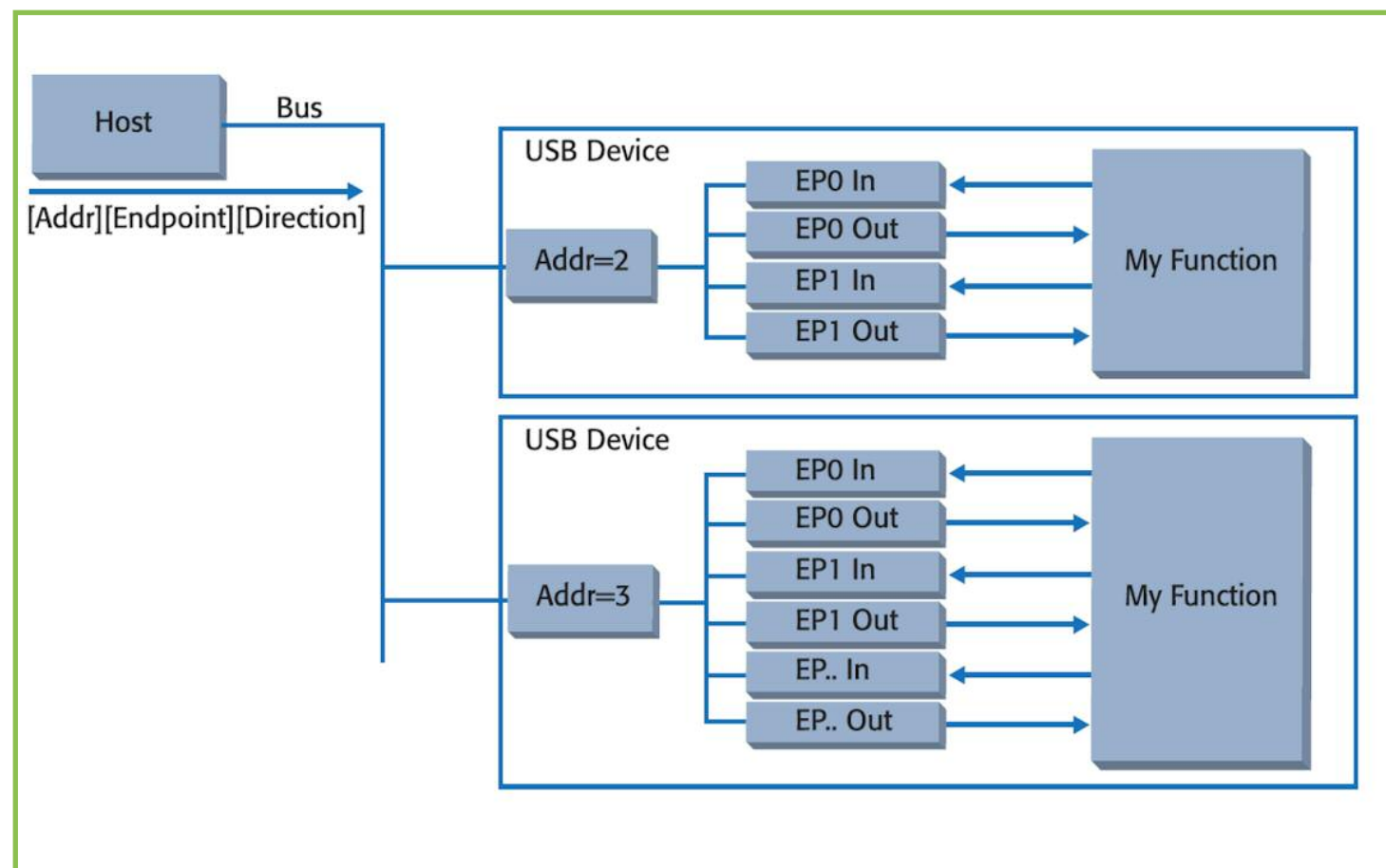



Figura 9 Schematizzazione degli endpoints

PIC18FX455/X550. Si può subito notare la presenta del regolatore di tensione a 3.3V interno, per consentire di utilizzare i resistori di pull-up on-chip, necessari per il riconoscimento della velocità. I due transistor MOS consentono di abilitare uno dei due resistori, a seconda della velocità desiderata. Il cuore di tutto il sistema è costituito dal SIE (Serial Interface Engine) che consente comunicazioni veloci tra il PIC e l'host. La scelta architetturale usata da Microchip consente di utilizzare sia il transceiver interno che uno esterno. In quest'ultimo caso bisogna utilizzare anche un regolatore a 3.3V e le relative resistenze. Per migliorare le performance è stata inserita una memoria dual port da 1KB (USB RAM) all'interno dell'area dati, come riportato in Figura 11. Per trasferire grandi quantità di dati può essere utilizzata l'interfaccia SSP (Streaming Serial Port). I registri necessari per la gestione ed il trasferimento con interfaccia USB sono di seguito elencati:

- USB Control register (UCON).
- USB ConFiGuration register (UCFG).
- USB Transfer Status register (USTAT).
- USB device ADDRess register (UADDR).
- Frame Number register (UFRMH: UFRML).
- Endpoint Enable register da 0 a 15 (UEPn).

Per ottimizzare la gestione della porta USB è stata introdotta una struttura logica per gli interrupt USB (figura 12), simile a quella già esistente per gli interrupt del PICMicro. In particolare, essi si suddividono in due livelli: le interruzioni dovute allo stato e quelle relative agli errori. Tutte le sorgenti di interrupt confluiscono, mediante porte AND e OR, in un'unica linea (USBIF), portata alla CPU. Per il funzionamento dell'hardware relativo all'USB è necessario un clock con frequenza pari a 48MHz.

LA PICDEM FS USB

Per ridurre il time-to-market nello sviluppo di sistemi embedded, Microchip ha reso disponibile un demo board: la PICDEM FS USB, di cui è mostrata un'immagine in figura 13. Essa è equipaggiata con il micro PIC18F4550 ed interfaccia USB 2.0 full speed. Le caratteristiche principali di questa EVK sono di seguito elencate:

- Frequenza di clock massimo 48MHz (12MIPS);
- 32KB di memoria programma;
- 2KB di RAM (di cui 1KB USB RAM di tipo dual port);
- 256bytes di EEPROM dati.

La scheda viene fornita con il bootloader precaricato; questo evita l'uso di un programmatore, poiché il download del firmware può essere eseguito semplicemente da seriale. Inoltre, sono resi disponibili gli schematici della scheda, figura 14, per eventuali sviluppi custom.

SIMULAZIONE DELL'INTERFACCIA SERIALE SU USB

Un'applicazione che può essere molto utile con l'interfaccia USB è l'emulazione di una porta seriale. La RS-232 è stata per molto tempo il principale metodo di comunicazione tra un PC ed un sistema embedded, anche grazie alla sua semplicità d'uso. Con l'avvento dei nuovi PC diventa sempre più raro trovare ancora una porta seriale. Una possibile soluzione a questo problema è la migrazione verso USB. Me-

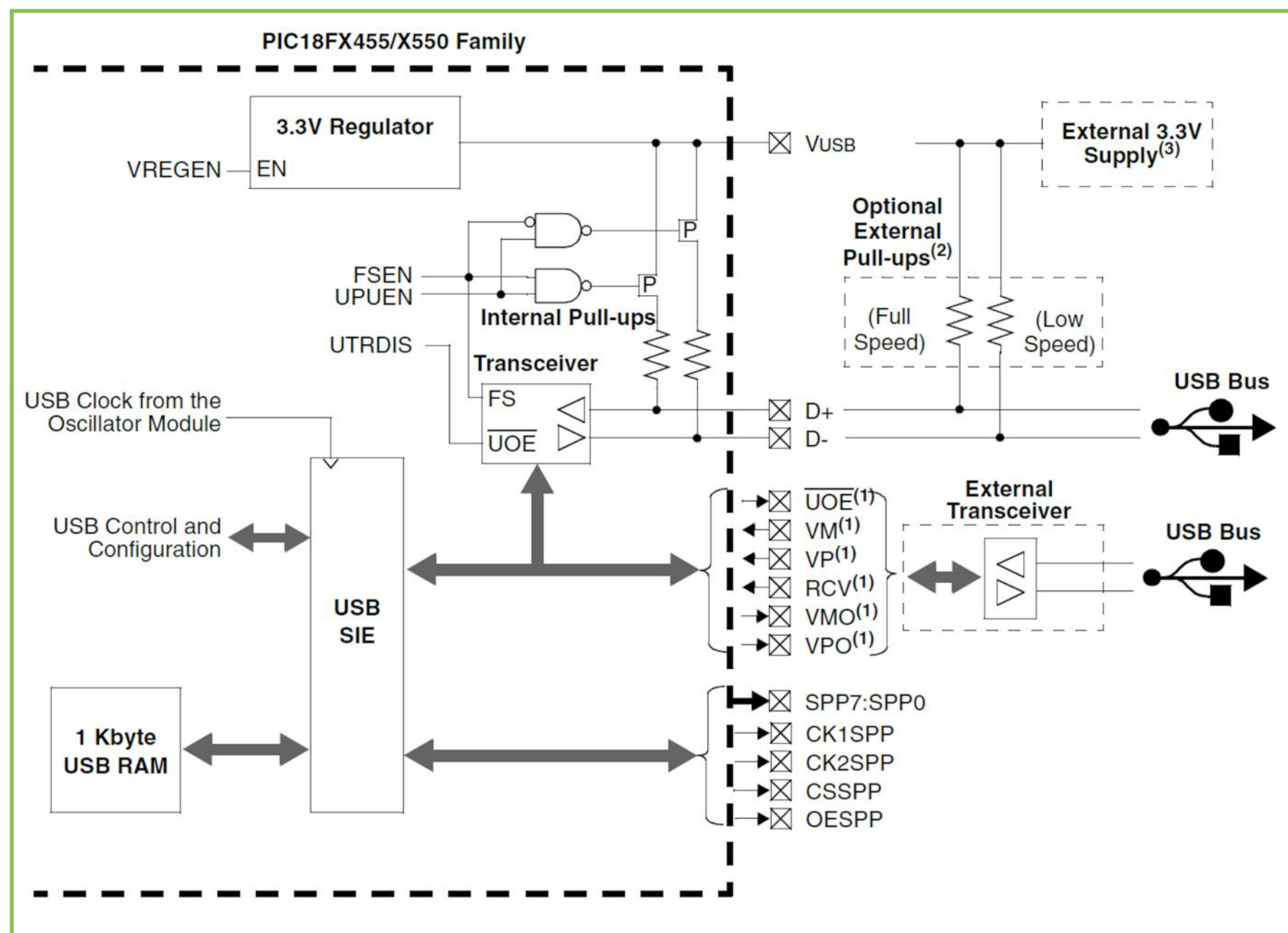



Figura 10 Schema completo della sezione USB

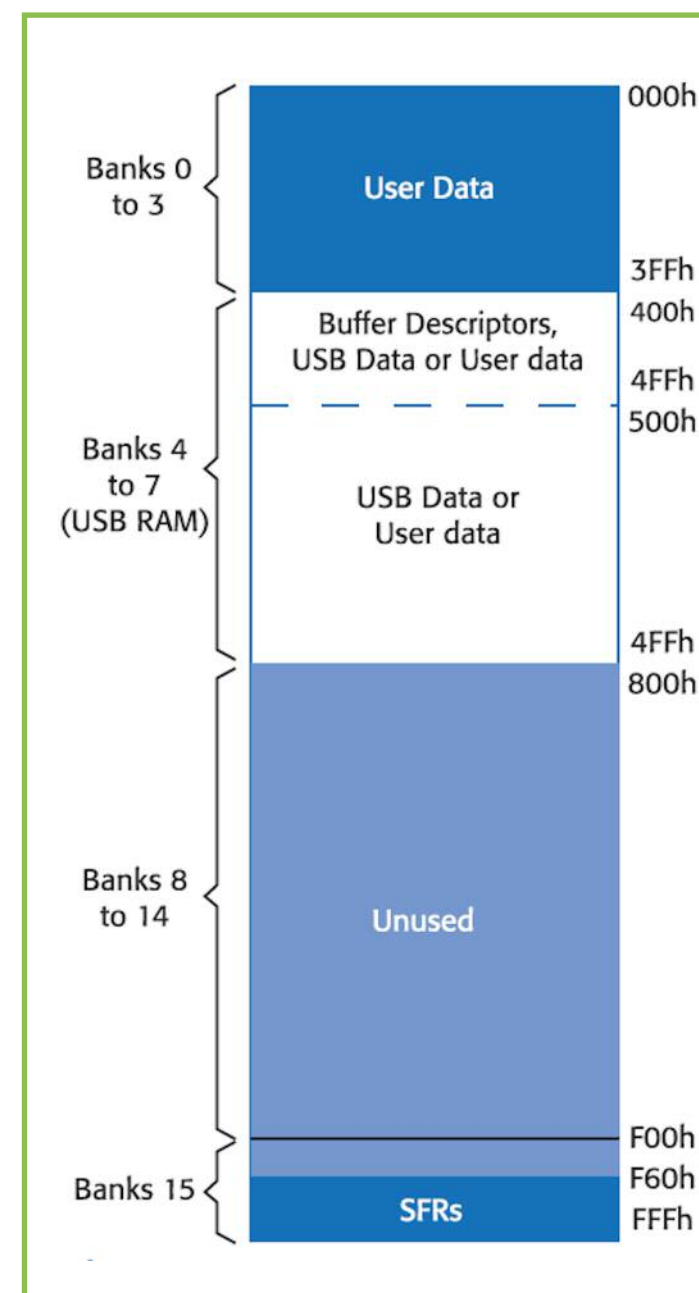



Figura 11 USB RAM

diante emulazione della seriale su USB il software continua a vedere un connessione RS-232 (quindi non è necessario modificarlo), mentre è necessario modificare leggermente l'hardware del sistema embedded. La figura 15 riporta uno schema della variazione architetturale che comporta il passaggio da RS-232 a USB. In rosso sono riportate le modifiche che è necessario apportare per adottare questa strada. Per quanto riguarda i driver CDC questi sono forniti con Windows oppure direttamente da Microchip. Per quanto riguarda l'hardware USB lato PICMicro, il problema è anche risolto poiché è sufficiente ricorrere ai modelli dei PIC elencati sopra. Rimane da progettare le funzioni (API) nel firmware del PICMicro per la gestione del protocollo USB. La buona notizia è che tali funzioni sono già stata sviluppate da Microchip (e riportate in tabella 4). Le funzioni sono scritte utilizzando il compilatore C18 e sono molto simili nel nome e nel funzionamento a quelle utilizzate per controllare la seriale. Si analizzeranno ora nel dettaglio alcune delle funzioni implementate e si cercherà far di comprendere come esse devono essere utilizzate per ottimizzare le prestazioni del firmware.

Invio di dati: la funzione "putrsUSBUSART"

Questa funzione scrive una stringa di dati verso porta USB, includendo il carattere di terminazione stringa (null character). Il dato da trasferire deve essere contenuto nella memoria programma, a differenza della funzione *putsUSBUSART* in cui il dato deve trovarsi nella RAM. Il prototipo della funzione è: `void putrsUSBUSART (const rom char *data)` dove data rappresenta un puntatore ad un stringa null-terminated. Di seguito sono riportati due esempi di utilizzo di questa funzione:

```
void example_1(void){
if(mUSBUSARTIsTxTrfReady())
putrsUSBUSART("Hello World.");
} //end example_1
rom char example_string[] =
{"Pippo"};
void example_2(void){
if(mUSBUSARTIsTxTrfReady())
putrsUSBUSART(example_string);
} //end example_2
```

FUNZIONE	DESCRIZIONE
putrsUSBUSART	Scrive una stringa con terminazione dalla memoria programma alla USB.
putsUSBUSART	Scrive una stringa con terminazione dalla memoria dati alla USB.
mUSBUSARTTxRom	Scrive una stringa di lunghezza prefissata dalla memoria programma alla USB.
mUSBUSARTTxRam	Scrive una stringa di lunghezza prefissata dalla memoria dati alla USB.
mUSBUSARTIsTxTrfReady	Verifica se il driver è pronto a ricevere una nuova stringa.
getsUSBUSART	Legge una stringa dalla USB.
mCDCGetRxLength	Legge la lunghezza dell'ultima stringa letta dalla USB.

Tabella 4

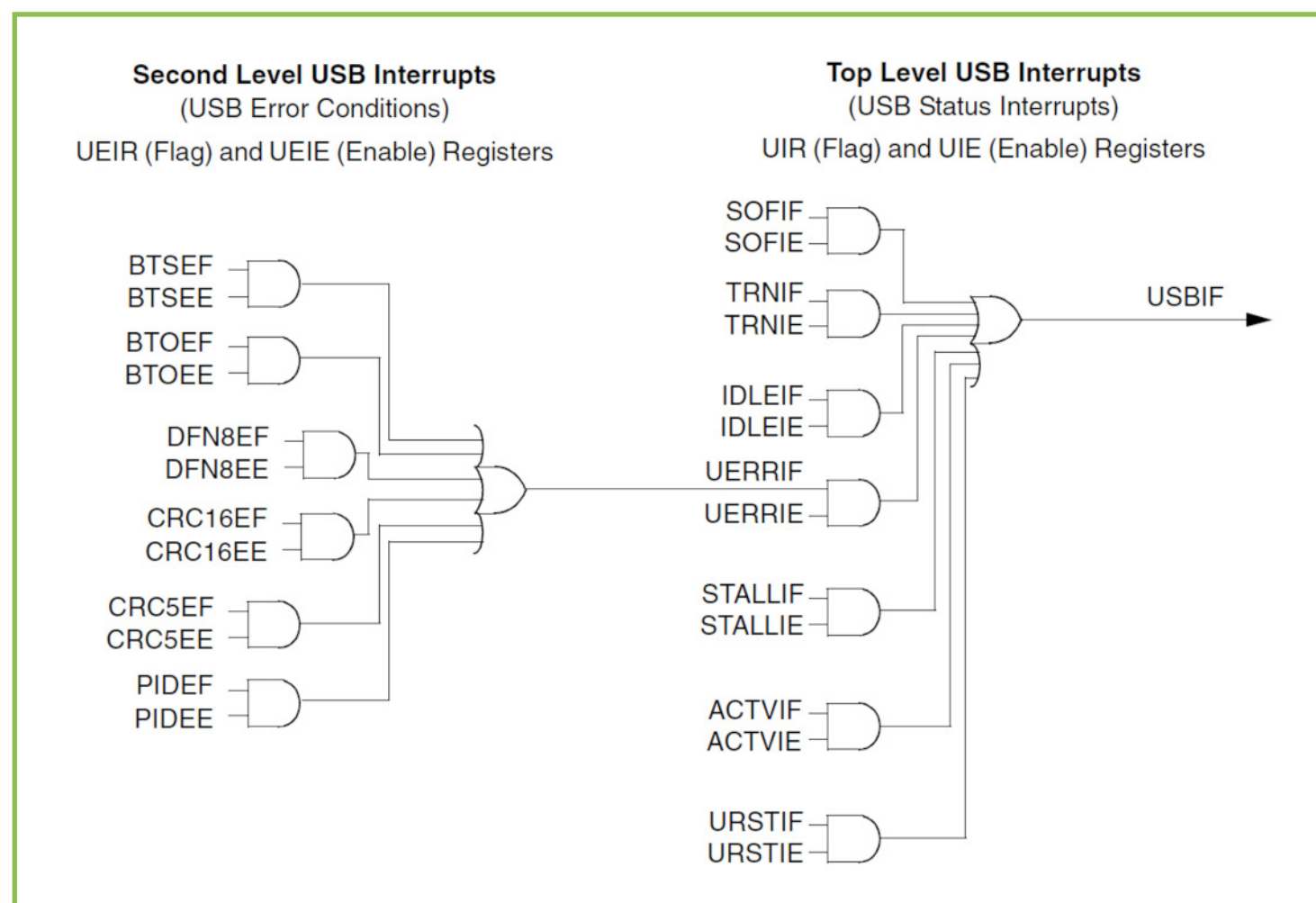



Figura 12 Logica degli interrupt USB

Figura 13 PICDEM FS USB

In entrambi gli esempi, oltre alla funzione putrsUSBUSART, è stata utilizzata mUSBUSARTIsTxTrfReady(). Essa restituisce 1 se il controller è pronto a ricevere altri dati. Prima di qualsiasi trasferimento è consigliabile utilizzarla onde evitare problemi di overbuffer dei dati.

Ricezione di dati: la funzione "getUSBUSART"

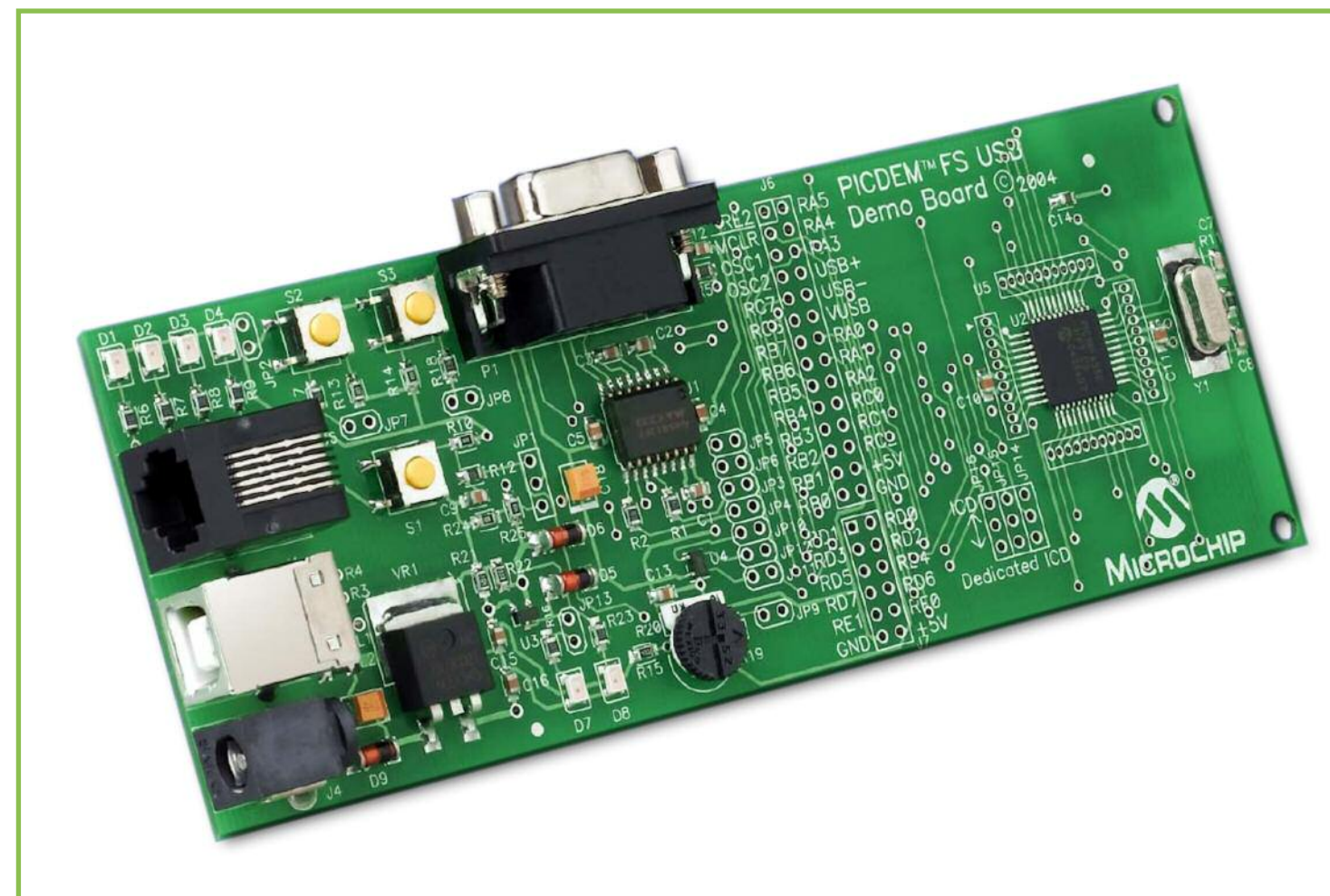
Questa funzione riceve una stringa dall'host USB e la trasferisce in una determinata locazione della RAM. Si tratta di un funzione cosiddetta "non bloccante" in quanto se non ci sono dati in ingresso non rimane in attesa ma restituisce 0 per segnalare tale condizione. La sintassi è di seguito riportata: `byte getUSBUSART (char *buffer, byte len)` dove `buffer` è la stringa in cui memorizzare i dati ricevuti, mentre len individua la lunghezza massima della stringa ricevuta. Se vengono ricevuti più byte di quelli indicati da len, allora solo i caratteri attesi saranno copiati nel buffer. Se invece accade di ricevere meno caratteri di quelli attesi, allora solo i caratteri ricevuti saranno inseriti in buffer. Il massimo valore di len deve essere inferiore alla massima dimensione dell'endpoint utilizzato per la ricezione dei dati. Per modificare questo valore bisogna intervenire sulla costante CDC_BULK_OUT_EP_SIZE all'interno del file *usbcfg.h*. Tale costante può assumere i valori 8,16, 32 oppure 64bytes. La funzione restituisce un valore di tipo byte, ossia il numero di caratteri ricevuti (0 indica che non sono stati ricevuti caratteri). Anche per questa funzione sono riportati due esempi di utilizzo:

```
char input_buffer[64];
void example_1(void){
byte index, count, total_sum;
if(getsUSBUSART(input_buffer,
8)){
count = mCDCGetRxLength();
total_sum = 0;
for(index = 0; index < count;
index++)
total_sum += input_buffer[in-
dex];
} //end if
} //end example_1
void example_2(void){
```

Figura 14 Schematico della PICDEM FS USB

```
if(getsUSBUSART(input_buffer,
CDC_BULK_OUT_EP_SIZE)){
// Do something…
} //end if
} //end example_2
```

Nel primo esempio è stata utilizzata un'altra funzione:
`mCDCGetRxLength()`.
Essa ha lo scopo di determinare il numero di caratteri ricevuti con la precedente chiamata della funzione
`getsUSBUSART(input_buffer, 8)`.

Osservazione sulle funzioni bloccanti

Ci sono alcune considerazioni da fare relativamente alla programmazione con le funzione USB/UART, per ottenere il massimo delle prestazioni:

- Il firmware fornito da Microchip è un ambiente multitasking di tipo cooperativo. Questo significa che non possono esserci funzioni bloccanti nel codice utente, poiché queste causerebbero lo stallo della macchina. In generale, conviene utilizzare una macchina a stati anziché funzioni bloccanti. Un esempio tipico di funzione bloccante è il seguente codice:
`while(!mUSBUSARTIsTxTrfReady());`
- Le funzioni `putrsUSBUSART`, `putsUSBUSART`, `mUSBUSARTTxRom` e `mUSBUSARTTxRam` sono non bloccanti. Si riportano ora due esempi di programmazione, rispettivamente incorretta e corretta, con le funzioni di cui sopra:

Esempio di programmazione errata:

```
if(mUSBUSARTIsTxTrfReady())
{
putrsUSBUSART("Hello World");
putrsUSBUSART("Hello Again");
}//end if
```

Esempio di programmazione corretta (macchina a stati):

```
byte state = 0;
if(state == 0){
if(mUSBUSARTIsTxTrfReady()){
putrsUSBUSART("Hello World");
state++;
} //end if
}
else if(state == 1){
if(mUSBUSARTIsTxTrfReady()){
putrsUSBUSART("Hello Again");
state++;
} //end if
} //end if
```

INSTALLAZIONE DEL SOFTWARE E DELL'HARDWARE

Come descritto in precedenza, un modo veloce per sviluppare le proprie applicazioni è quello di fare uso della PICDEM FS USB. È possibile effettuare il download dal sito Microchip del software di gestione, all'indirizzo:
http://ww1.microchip.com/downloads/en/AppNotes/ CDC_RS232_Emulation.EXE.
Se sono stati utilizzati i parametri di default, il percorso di installazione è C:\MCHPFSUSB\. Se si collega la PICDEM al PC mediante il cavo USB si nota la tipica schermata di quanto viene rilevata una periferica USB e bisogna specificare il percorso del driver (figura 16). Il nome del driver è mchpcdc.inf e si trova al seguente percorso:
C:\MCHPFSUSB\fw\Cdc\inf\win2k_winxp.
Dopo aver completato l'installazione del

Figura 15 Migrazione da RS-232 a USB

driver si nota, andando nella finestra *Gestione Periferiche*, che è comparsa un'altra COM (porta seriale virtuale). Il lato PC a questo punto è pronto. Prima di utilizzare il file .inf nei propri progetti è necessario cambiare le impostazioni di default, tra cui VID e PID. È sufficiente aprire il file con un editor di testi (anche blocco note) e ricercare la stringa "USB\VID_xxxx&PIDyyyy" dove "xxxx" è il valore esadecimale del VID e "yyyy" il valore esadecimale del PID; generalmente la stringa è contenuta sotto l'header [DeviceList]. Oltre a queste modifiche è possibile cambiare i parametri riportati sotto l'header [Strings]; quelli riportati di default sono:

- MCHP="Microchip Technology Inc."
- MFGNAME="Microchip Technology Inc."
- DESCRIPTION="Communications Port"
- SERVICE="USB RS-232 Emulation Driver"

Resta da configurare il lato micro. Per le prime prove è possibile utilizzare il progetto già pronto C:\MCHPFSUSB\fw\Cdc\MCHPUSB.mcp . Esso consiste di diversi files tra cui il già citato *usbcfg.h* ed il *cdc.c* che contiene invece tutte le funzioni USB/UART. Per la realizzazione di un progetto custom, invece, è necessario:

1. Inserire *#include "system\usb\usb.h"* in ciascun file che utilizza le funzioni in oggetto.

2. Definire la costante USB_USE_CDC all'interno del file *usbcfg.h*.

3. I file *cdc.c* e *cdc.h* devono essere aggiunti al pro getto. Essi si trovano al percorso: C:\MCHPFSUSB\fw\Cdc\system\usb\class\cdc

CHIP DEDICATI

Per la conversione seriale/USB con USB slave esistono anche chip dedicati che da un lato forniscono una connettività USB, dall'altro rendono disponibile una (o più) porte seriali standard. È il caso del chip MCS7703 prodotto da Moschip Semiconductor di cui si riportano le caratteristiche fondamentali:

- Alimentazione a 5V con regolatore interno a 3,3V;
- Certificazione WHQL;
- Data rates supportati: da 75mbps fino a 3Mbps;
- USB1.1 compliant;
- VID e PIC riconfigurabili mediante EEPROM esterna;

La figura 17 mostra lo schema a blocchi ed il pinout del dispositivo.

Il vantaggio nell'usare un chip del genere risiede principalmente nel fatto che i driver vengono forniti direttamente dal costruttore e l'operazione di conversione avviene in modo del tutto trasparente all'utilizzatore.

FTDI FT232R: una soluzione alternativa

Un'altra possibilità per la conversione USB/seriale è offerta dal nuovo chip della FTDI, FT232R, che rappresenta alcune im-

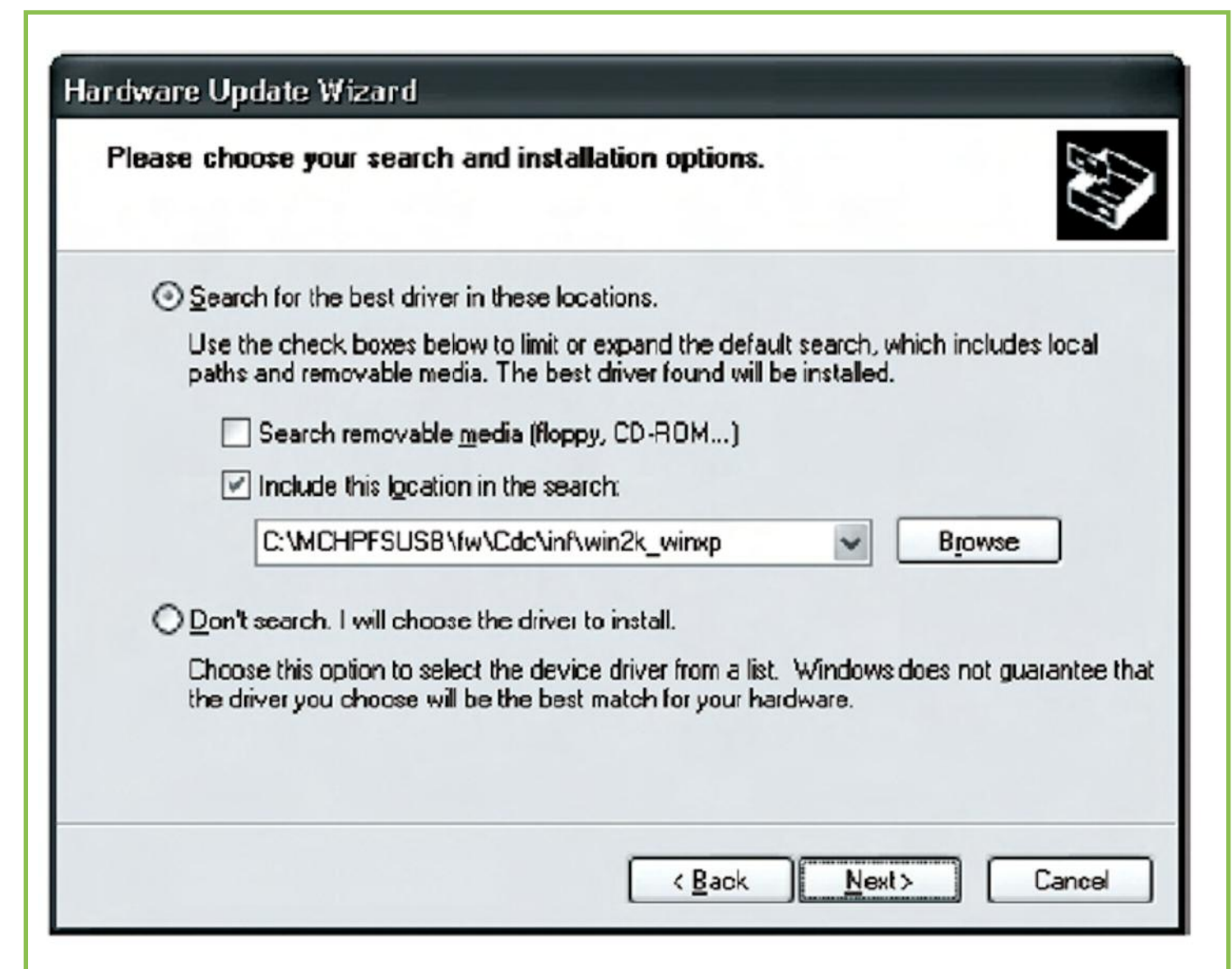



Figura 16 Richiesta del driver CDC RS-232 da parte del S.O

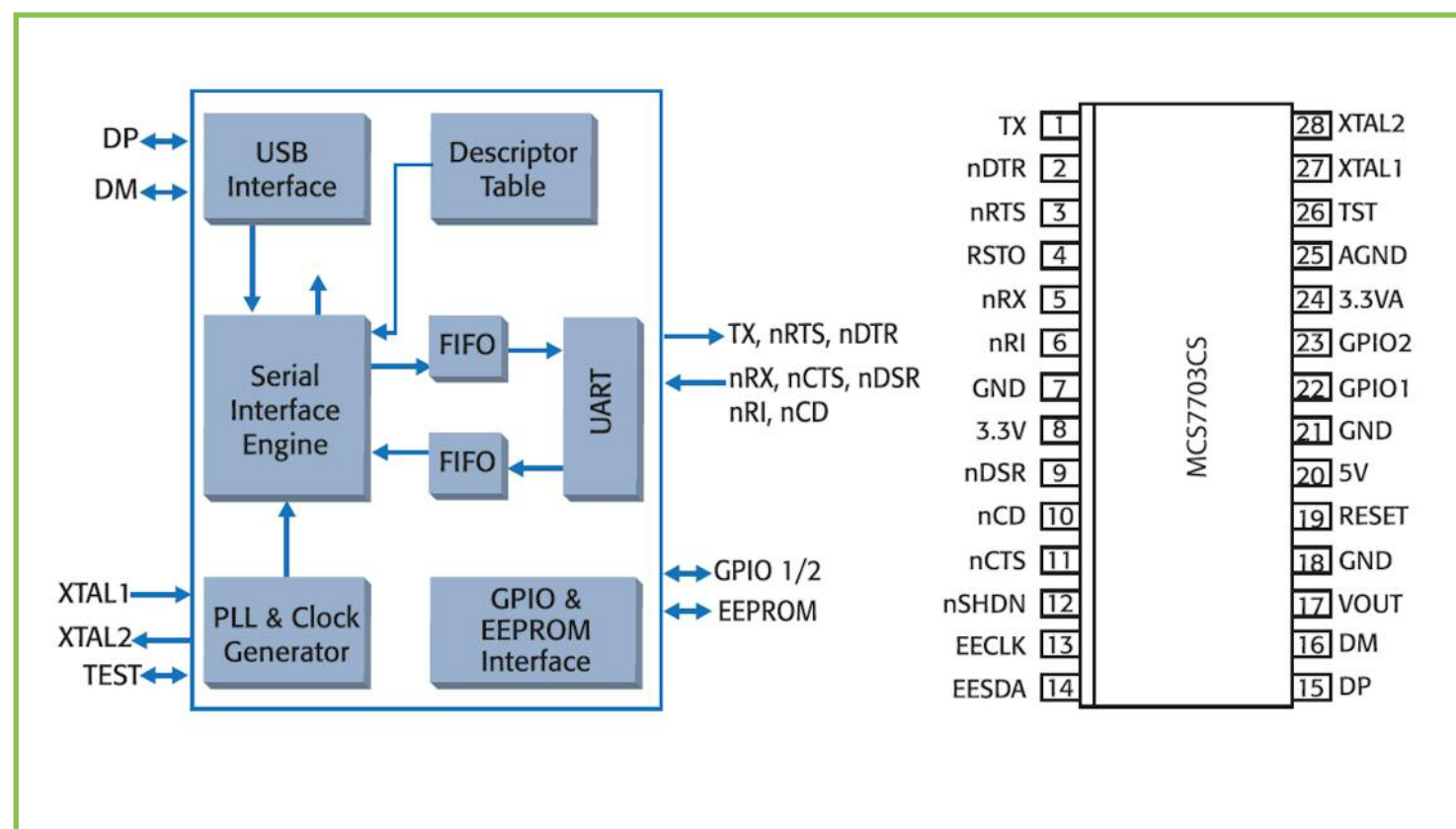



Figura 17 Schema a blocchi e pinout del convertitore seriale/USB MCS7703

portanti novità rispetto al suo predecessore FT232BM:

1. Tutti i componenti chiave sono stati integrati nel chip (tra cui la EEPROM, il clock configurabile ed altri componenti passivi).

2. L'identificativo del dispositivo fornito da FTDI è univoco.

3. Possibilità di ridurre fino al 50% del area riservata al convertitore.

4. Due differenti package: 28-pin SSOP (FT232RL) and 32-pin QFN (FT232RQ)

La Figura 18 riporta lo schema a blocchi di questo chip. Inoltre, il produttore riporta tutti gli schematici per implementare il convertitore USB/Seriale.

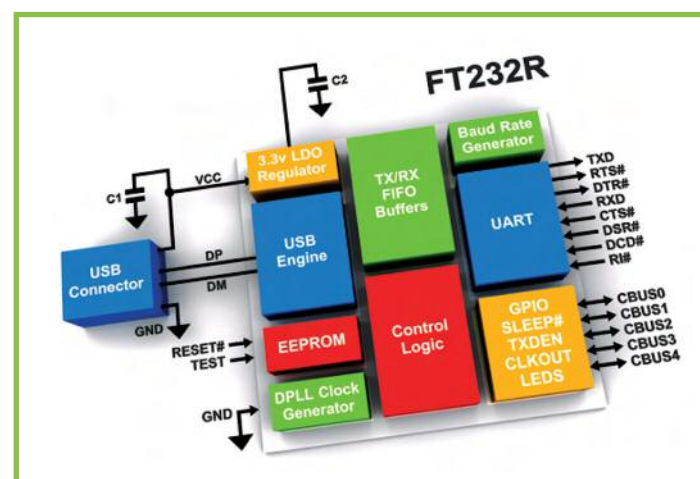



Figura 18 Schema a blocchi dell'FT232R

Per approfondire

[1] "PIC18F2455/2550/4455/4550 Data Sheet" (DS39632)"

[2] "USB Specification Revision 2.0", USB Implementers

[3] "USB Class Definitions for Communication Devices Version 1.1", USB Implementers Forum Inc., 1999.

[4] "Migrating Applications to USB from RS-232 UART with Minimal Impact on PC Software" – AN956

di MAURIZIO DEL CORSO

CONTROLLO MOTORE CON DSPIC

Un progetto per pilotare un motore o un qualsiasi carico induttivo sfruttando la periferica PWM del dsPIC

Il PWM, acronimo di *Pulse Width Modulation*, è una tecnica per il controllo di dispositivi analogici mediante segnali digitali generati tipicamente da un microcontrollore. La tecnica consiste nel modulare il duty cycle di un'onda quadra in accordo ad un segnale di controllo: il risultato è quindi una sequenza di impulsi aventi tutti la medesima ampiezza ma aventi una durata diversa in funzione di un segnale di controllo e l'effetto è quello di alimentare il carico per intervalli di tempo finiti in modo da controllarne la potenza assorbita. Uno dei vantaggi di questa tecnica è quello che il segnale di controllo che pilota direttamente il carico è digitale, quindi non è necessario operare conversioni DA, l'effetto del rumore è decisamente marginale e la lunghezza del collegamento fisico tra con trollore e carico può essere anche significativa senza compromettere l'integrità del segnale. In prossimità del carico è sufficiente un filtro passabasso passivo per rimuovere le armoniche ad alta frequenza del segnale PWM.

Figura 1 La struttura dell'unità Motor Control PWM del dsPIC30F

IL MODULO PWM

Nel dsPIC30F è previsto un timer a 16 bit dedicato esclusivamente alla generazione della base dei tempi per i segnali PWM. Nella figura 1 è riportato lo schema a bloc-

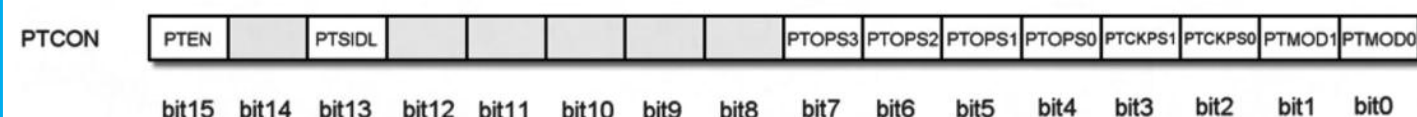



Figura 2 La struttura del registro PTCON

chi della periferica PWM del dsPIC30F. Si noti che esiste un'unica base dei tempi e quattro diversi blocchi di generazione del duty cycle: questo permette di controllare 4 periferiche indipendenti. La base dei tempi viene generata dal timer PTMR al quale è possibile collegare un prescaler in modo da poter dividere la frequenza base (al massimo pari alla frequenza di un ciclo istruzione) per 1, 4, 16 o 64. Il timer viene automaticamente incrementato o decrementato (a seconda del valore del bit 15 del PTMR) e comparato con il valore impostato per il periodo (scritto nel registro PTPER) quindi viene generata una interruzione quando i due valori coincidono. Il modulo può anche essere configurato per generare interruzioni in corrispondenza di più occorrenze di uguaglianza tra PTMR e PTPER ed in particolare si può arrivare alla generazione di una interruzione in corrispondenza della sedicesima volta che i valori di PTMR e PTPER coincidono. Il controllo e la configurazione del modulo PWM avviene per mezzo del registro PTCON la cui struttura è riportata nella figura 2. In particolare mediante questo registro è possibile abilitare/disabilitare il modulo (bit 15) impostare il prescaler/postscaler e scegliere uno dei quattro modi di funzionamento della periferica: *free running, single event, continuos up/down, continuos up/down with double PWM update*.

Modalità free running

Operando in modalità *free running* è possibile ottenere uscite PWM allineate al fronte di salita (*Edge Aligned PWM*). Si consideri la figura 3a in cui è riportato l'incremento del PTMR ed i livelli per PTPER, PDC1 e PDC2 relativi alle uscite PWM1 e PWM2. Inizialmente tutte le uscite sono attive (in figura sono riportare per semplicità le sole due uscite attive alte PWM1H e PWM2H) e quando il valore di PTMR coincide con quello di PDC1 (che fissa il duty cycle per l'uscita PWM1) l'uscita PWM1 commuta a livello basso. La stessa cosa avviene in maniera analoga per l'uscita PWM2 quando il PTMR coincide con PDC2. A livello firmware è possibile aggiornare il duty cycle in qualsiasi momento, ma operativamente i registri PDC vengono aggiornati con i nuovi valori solo all'inizio di un nuovo ciclo (ovvero quando PTMR è zero). Se le interruzioni sono abilitate si ha una interruzione all'inizio di ogni ciclo.

Modalità Single Event

La figura 3b mostra il segnale PWM ottenuto utilizzando la modalità *single event*. Questa modalità prevede un solo ciclo di

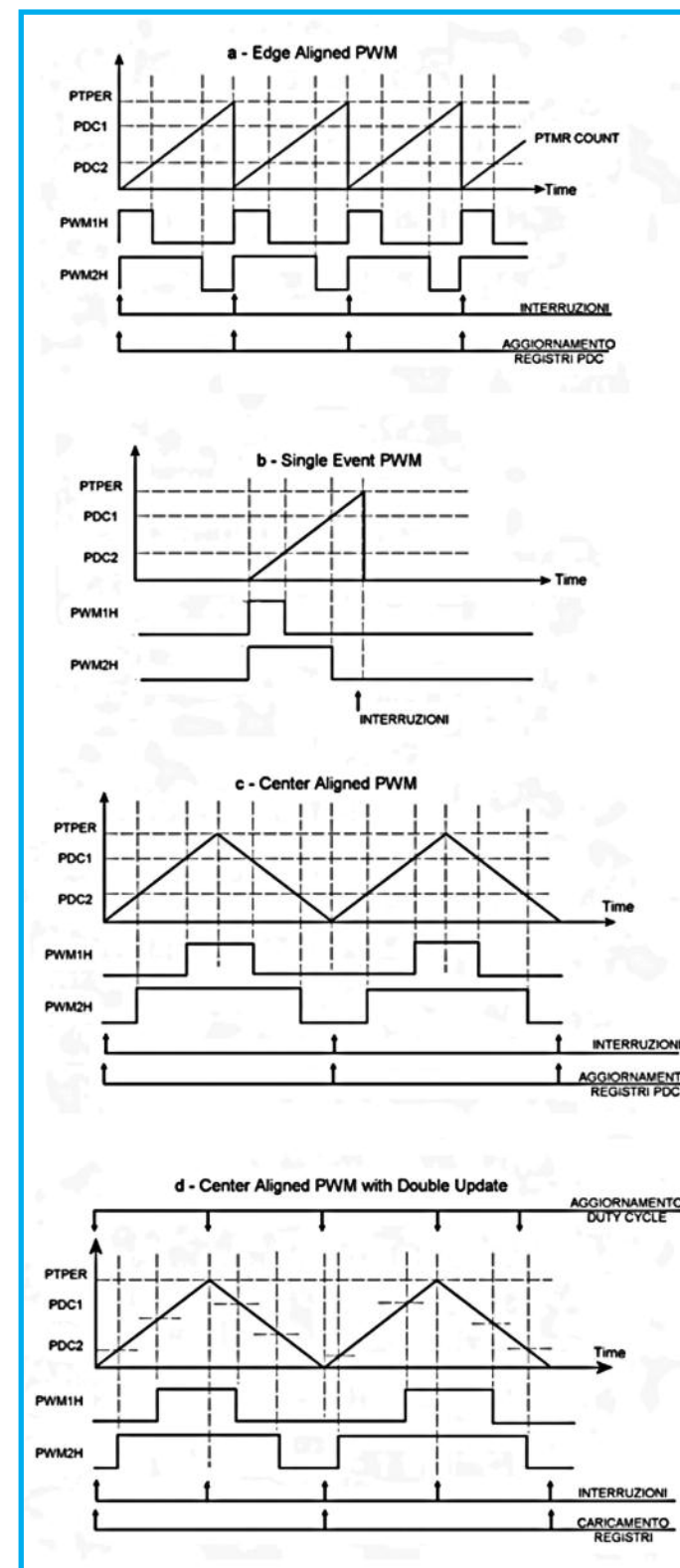



Figura 3 Le varie modalità di funzionamento del modulo PWM

conteggio per PTMR (il bit PTEN si azzera automaticamente all'azzeramento del PTMR) per cui viene generato un singolo impulso PWM sulle varie uscite. In questo caso, abilitando le interruzioni, viene generata una unica interruzione alla fine del ciclo.

Modalità continuos up/down

In questa modalità è possibile ottenere uscite PWM i cui impulsi sono tutti centrati, come mostrato nella figura 3c. In questo caso il PTMR funziona in modalità up/down per cui il periodo del segnale PWM risulta il doppio del valore impostato in PTMR. A differenza della modalità *free running* in questo caso si evita la commutazione simultanea di più uscite quindi consente di limitare i picchi di assorbimento inevitabilmente presenti durante le fasi di commutazione. Le interruzioni, se abilitate, vengono generate in cor rispondenza di ogni azzeramento del PTMR e nello stesso istante, viene anche aggiornato il contenuto dei registri PDC.

Modalità continuos up/down with double update

Questa modalità, il cui funzionamento è illustrato in figura 3d, è molto simile alla precedente con l'unica differenza che i registri PDC vengono aggiornati sia all'azzeramento di PTMR sia quando lo stesso timer raggiunge il massimo valore (impostato mediante PTPER). Utilizzando questa tecnica è possibile ottenere segnali PWM non simmetrici.

L'unità Dead Time

Utilizzando i due segnali complementari di una stessa uscita PWM (ad esempio PWM1H e PWM1L) entra in gioco l'unità *Dead Time*. Utilizzando infatti i due segnali complementari per una applicazione tipo quella di figura 4, c'è il rischio che, a causa di asimmetrie sul tempo di salita/discesa dei fronti del segnale, le due uscite si trovino entrambe attive in uno stesso

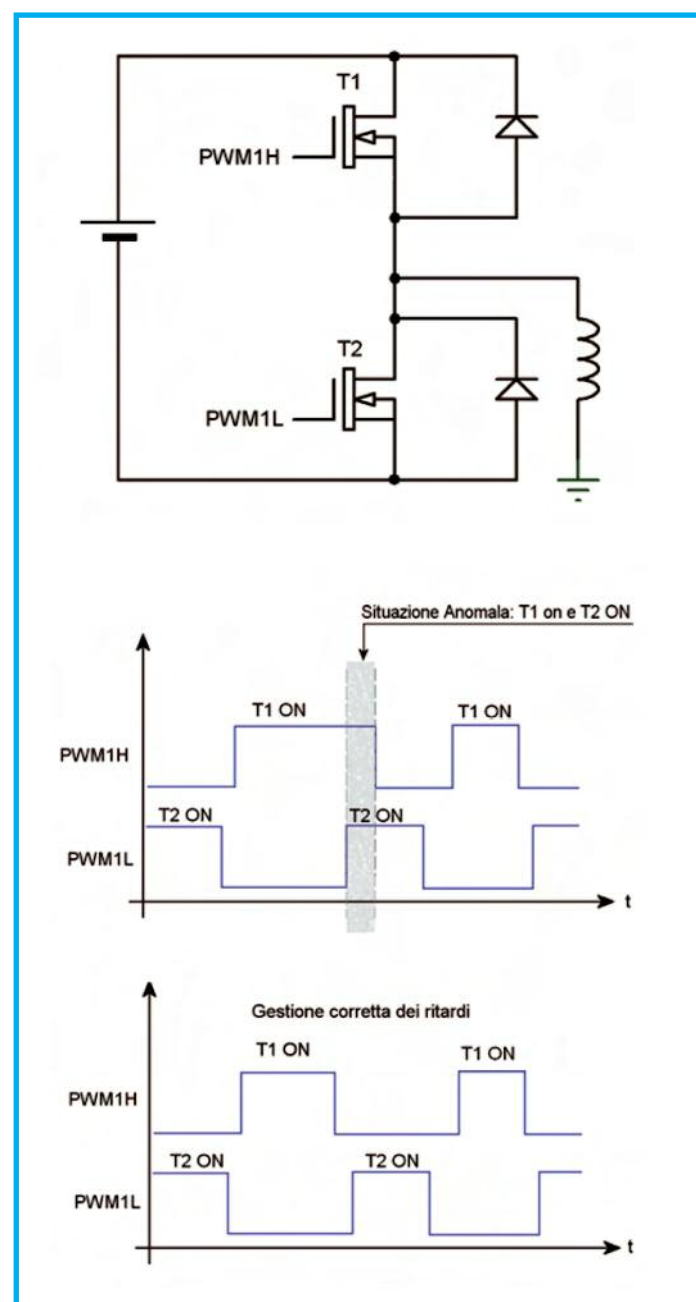



Figura 4 L'utilità del blocco Dead Time

Figura 5 Struttura dei registri DTCON1 e DTCON2

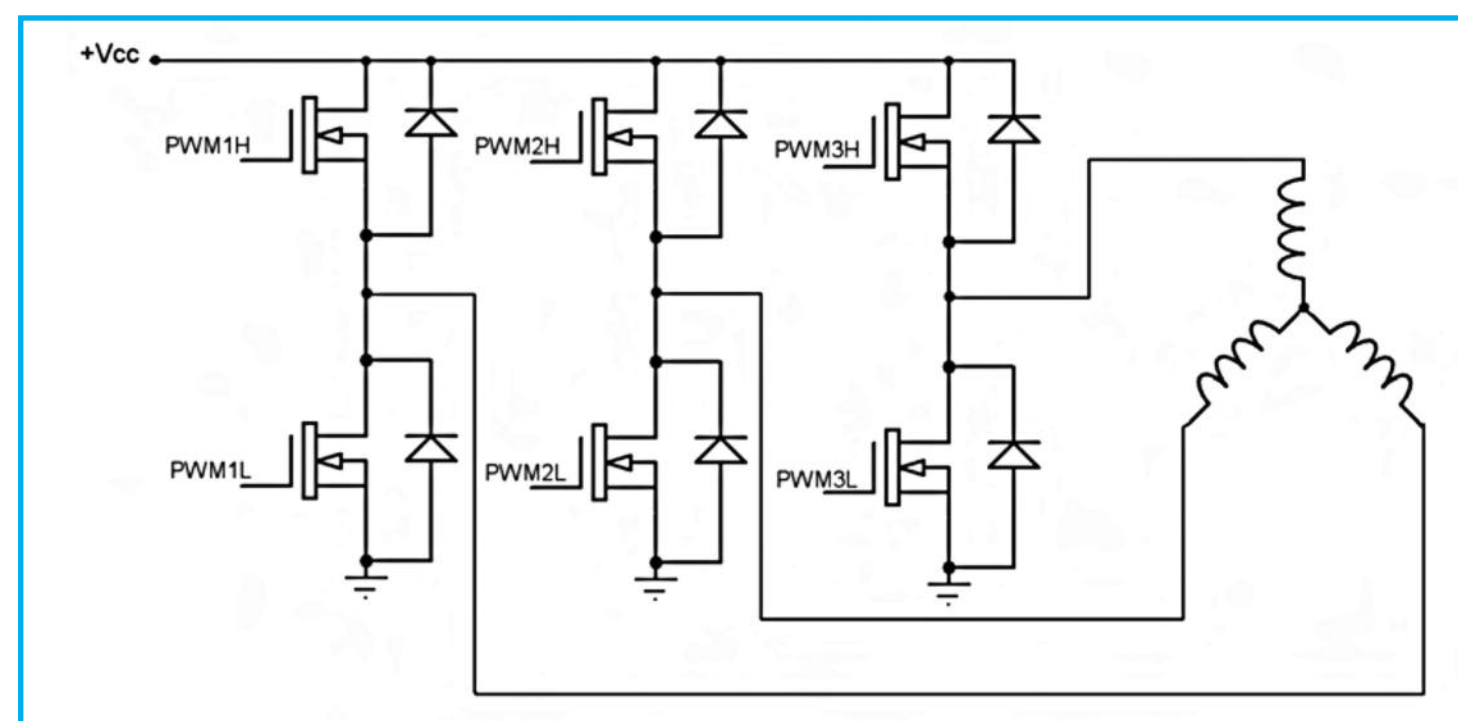



Figura 6 Pilotaggio di un carico trifase con uscite complementari

istante provocando un corto-circuito sulla linea di alimentazione. L'unità *Dead Time* inserisce un piccolo ritardo nella commutazione dei segnali in modo che questi non risultino mai attivi contemporaneamente. Nei dispositivi con 8 uscite PWM sono disponibili due valori per la *dead time* (A e B): questo consente di compensare anche eventuali differenze nei tempi di commutazione della coppia di transistori di potenza e, allo stesso tempo, poter gestire differenti stadi di pilotaggio con lo stesso chip.

I registri coinvolti nella configurazione dell'unità *Dead Time* sono DTCON1 e DTCON2 la cui struttura è riportata in figura 5. I ritardi sono calcolati mediante due contatori a 6 bit il cui valore viene confrontato con il limite impostato nel campo DTA e DTB di DTCON1. I bit DTxPSy (bit 6,7 e 14,15) permettono di impostare un prescaler per il timer associato all'unità fornendo all'utente la possibilità di impostare un fattore di divisione pari a 2, 4 o 8 rispetto alla massima frequenza (frequenza dei cicli istruzione). DTCON2 permette invece di associare una delle due unità (A o B) ad ogni uscita e su ciascuno dei due fronti (salita/discesa).

TECNICHE DI PILOTAGGIO DEI CARICHI

Uscite complementari

In figura 6 è riportata una applicazione in cui viene pilotato un carico trifase in PWM sfruttando tre coppie di uscite complementari del dsPIC30F. Ciascuna fase è pilotata da una coppia di MOS di potenza che vengono gestiti direttamente dalle uscite PWM. In questa applicazione ciascuna uscita di ogni coppia può essere pilotata in maniera indipendente o in maniera complementare.
Nel primo caso l'unità *Dead Time* viene automaticamente disabilitata e l'utente ha il pieno controllo dello stato delle uscite. Nel secondo caso la coppia è pilotata in modo complementare e l'unità *Dead Time* provvede ad inserire automaticamente i ritardi opportuni affinché i due MOS non si trovino mai contemporaneamente in conduzione.

Uscite indipendenti

La figura 7 mostra il pilotaggio di un motore a quattro fasi a riluttanza variabile in cui ciascuna coppia di uscite pilota una singola fase del motore in modo indipendente. Essendo indipendenti, le singole uscite di una stessa coppia possono essere attivate anche contemporaneamente (condizione necessaria per polarizzare gli avvolgimenti di ciascuna fase).

GESTIONE PWM IN MIKROBASIC PER DSPIC

Come esempio di applicazione del controllo in PWM in linguaggio ad alto livello verranno illustrate alcune routines scritte in mikroBASIC per dsPIC. Le funzioni della libreria PWM sono cinque ed in particolare:

```
PWM_Init
PWM_Start
PWM_Set_Duty
PWM_Stop
PWM_Calc_Period
```

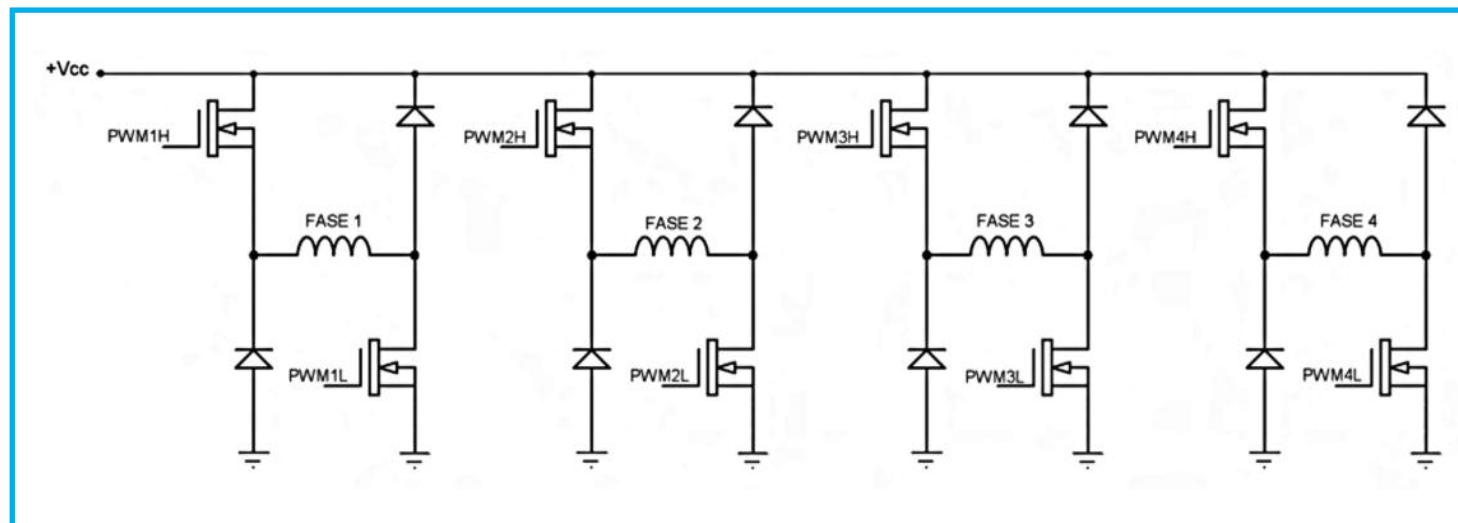



Figura 7 Un esempio di pilotaggio di motore a riluttanza variabile a 4 fasi

La `PWM_Init` inizializza il modulo PWM e richiede quattro parametri: la frequenza desiderata (in Hertz) per il segnale PWM, la modalità di funzionamento delle quattro uscite (indipendenti o complementari), l'abilitazione delle uscite e la configurazione del prescaler/postscaler. Per la modalità di funzionamento si devono specificare 4 bit (uno per ciascuna uscita) e il valore 0

NIBBLE	PRESCALER	POSTSCALER
0	1:1	1:1
1	1:2	1:4
2	1:3	1:16
3	1:4	1:64
4	1:5	-
5	1:6	-
6	1:7	-
7	1:8	-
8	1:9	-
9	1:10	-
10	1:11	-
11	1:12	-
12	1:13	-
13	1:14	-
14	1:15	-
15	1:16	-

Figura 8 le impostazioni del prescaler/postscaler

corrisponde alla modalità indipendente mentre il valore 1 a quella complementare. Il parametro per l'abilitazione delle uscite consiste in un byte in cui ciascun bit corrisponde ad una singola uscita:

`<pwm4h.pwm3h.pwm2h.pwn1h.pwm4l.pwm3l.pwm2l.pwm1l>`

Se un bit è a zero la relativa uscita è disabilitata (quindi funziona come I/O generico) altrimenti è una uscita PWM. Il parametro relativo al prescaler/postscaler consta di un byte i cui 4 bit meno significativi sono relativi al postscaler mentre i 4 più significativi al prescaler. La corrispondenza tra il valore del nibble e l'impostazione del

Figura 8 le impostazioni del prescaler/postscaler

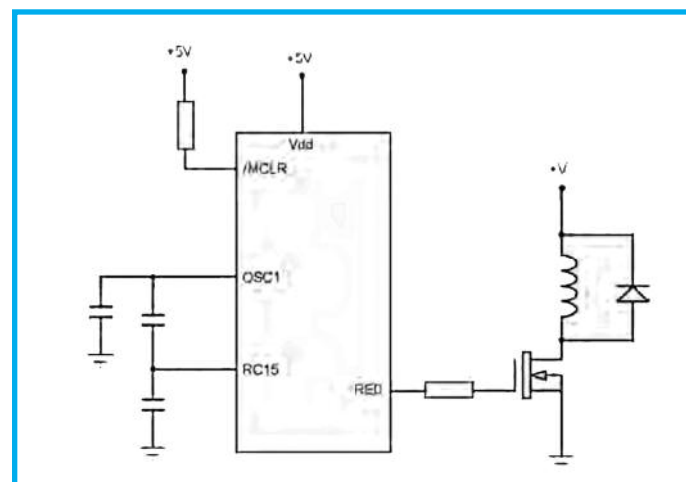
LISTATO 1

Download

```
Program Controllo_PWM
dim duty_cycle as word
main:
duty_cycle=0x1FF
'attesa di 1 sec prima di avviare il motore
Delay_ms(1000)
'inizializzazione del modulo PWM:
' frequenza: 3KHz
' modalità: uscite complementari (per tutti i canali)
' Abilitazione: tutti i canali abilitati
' pre-postscale: prescaler 1:1 postscaler 1:1
PWM_Init(3000, %1111, 0xFF, 0x00)
'impostazione del duty cycle per l'uscita 1
PWM_Set_Duty(0x000F,1)
'avvio del modulo PWM: partenza del motore
PWM_Start()
'variazione della velocità del motore (ciclo infinito)
while TRUE
PWM_Set_Duty(duty_cycle,1)
Delay_ms(10)
Inc(duty_cycle)
if duty_cycle >= 0x4FF then
duty_cycle = 0xF
end if
wend
end.
```

prescaler/postscaler è riportata nella tabella di figura 8. La `PWM_Set_Duty` permette di aggiornare il duty cycle per l'uscita desiderata. I parametri sono due word in cui la prima specifica il valore e la seconda il canale. `PWM_Start` avvia la generazione dei segnali PWM e non necessita di alcun parametro. Analogamente la `PWM_Stop` arresta il modulo PWM. Infine la `PWM_Calc_Period` permette di calcolare il corretto valore per il registro PTPER passando alla funzione la frequenza (in Hertz) ed il prescaler. Il calcolo del valore da inserire in PTPER viene effettuato secondo la formula PTPER=-1+F_{CY}(F_{PWM}*Prescaler) dove FCY è la frequenza del ciclo istruzione e FPWM è la frequenza desiderata per il segnale PWM.

Un esempio pratico

Come esempio pratico di applicazione in PWM si consideri lo schema di figura 9 in cui un dsPIC30F2010 pilota un motore DC monofase mediante un MOSFET di potenza. La routine di esempio (si veda il listato 1) inizializza il modulo PWM e pilota automaticamente il motore regolandone ciclicamente la velocità da un minimo ad un massimo. La regolazione della velocità viene fatta aumentando il duty cycle il cui valore è contenuto nella variabile duty_cycle.

di GIUSEPPE AQUILANI (IK0JREE)

GENERATORE DI SEGNALI

Costruiamo un generatore di onde quadre e sinusoidali con microcontrollore e display LCD

In questi ultimi periodi sono stati sviluppati dei circuiti basati su circuiti sintetizzatori della serie AD9X, in particolare gli AD9850 e AD9851 prodotti dalla Analog Devices (http://www.analog.com) che permettono allo sviluppatore di realizzare circuiti per generatori di segnale con pochissimi componenti aggiuntivi.
In particolare i due sintetizzatori citati sono in grado di digitalizzare una determinata frequenza all'interno del loro range, partendo da un'informazione digitale ovvero elaborata e fornita da un microcontrollore o microprocessore. In particolare , ad ogni frequenza da generare, il sintetizzatore fa corrispondere una "tuning word", calcolata con l' apposita formula descritta nel datasheet degli AD9XX: fout=((tuning word)*CLCKIN)/2^32, dove fout è la frequenza desiderata in Mhz, "tuning word" il valore a 32bit da inviare all'AD e CLKIN la frequenza in Mhz del clock dell'AD.
Tale "tuning word", inviata al sintetizzatore, fa sì che quest'ultimo produca (sintetizzi) in uscita un'onda sinusoidale e/o quadra corrispondente alla frequenza desiderata.
NOTA: la "tuning word" è una cosiddetta "frase" composta da 32 bit. Nel linguaggio tecnico di programmazione, la "word" è nor-

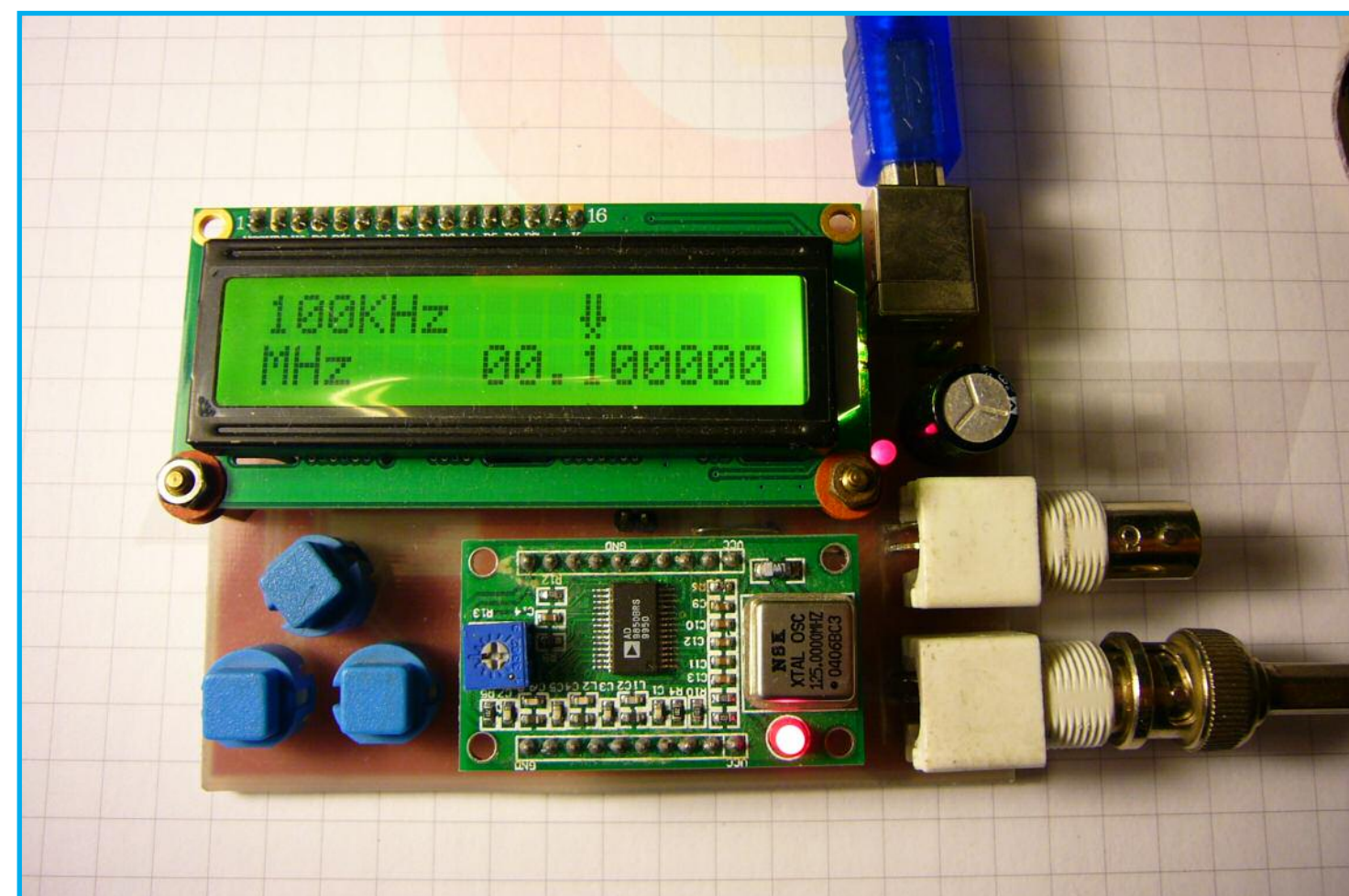



Figura 1

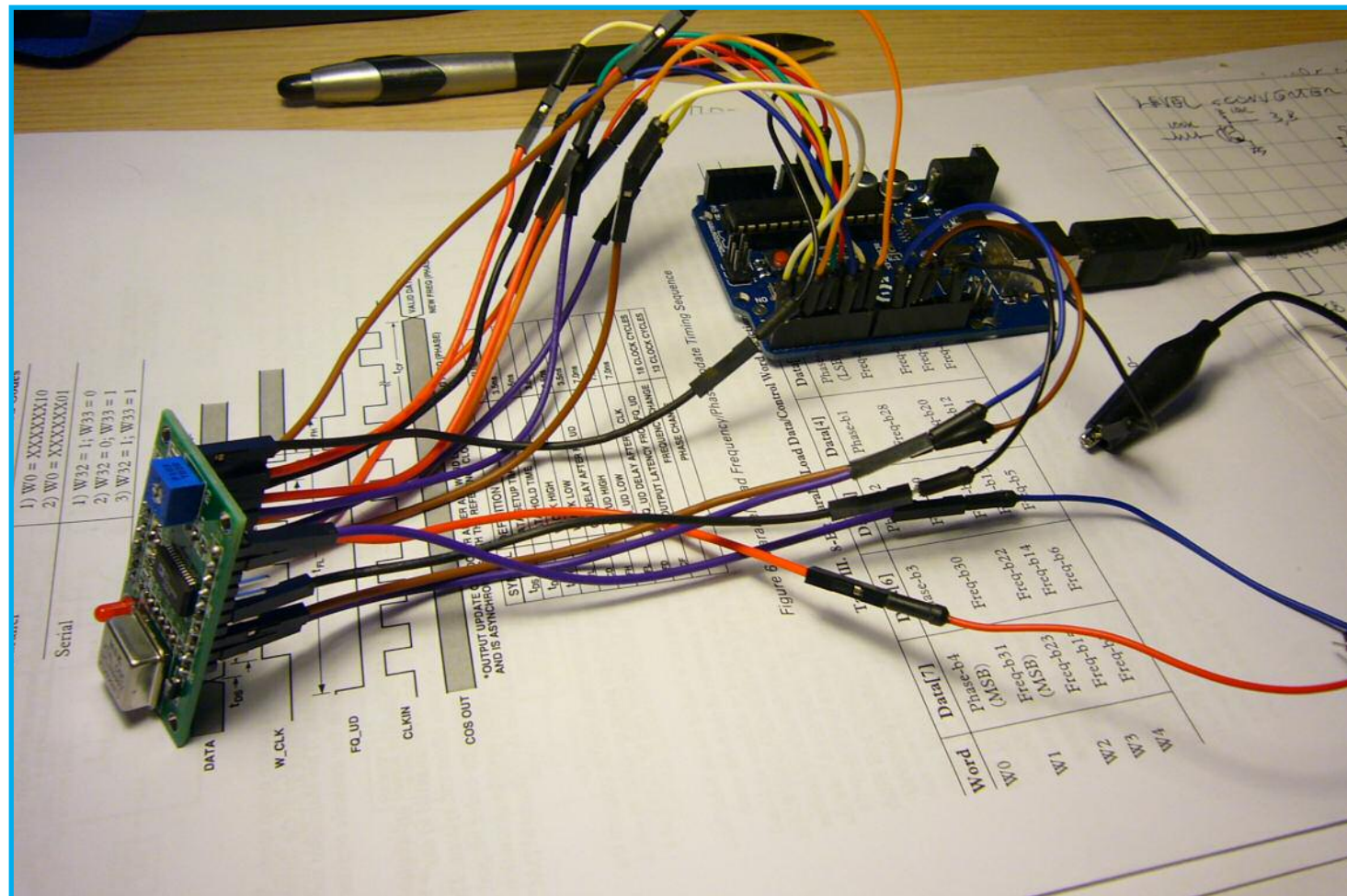
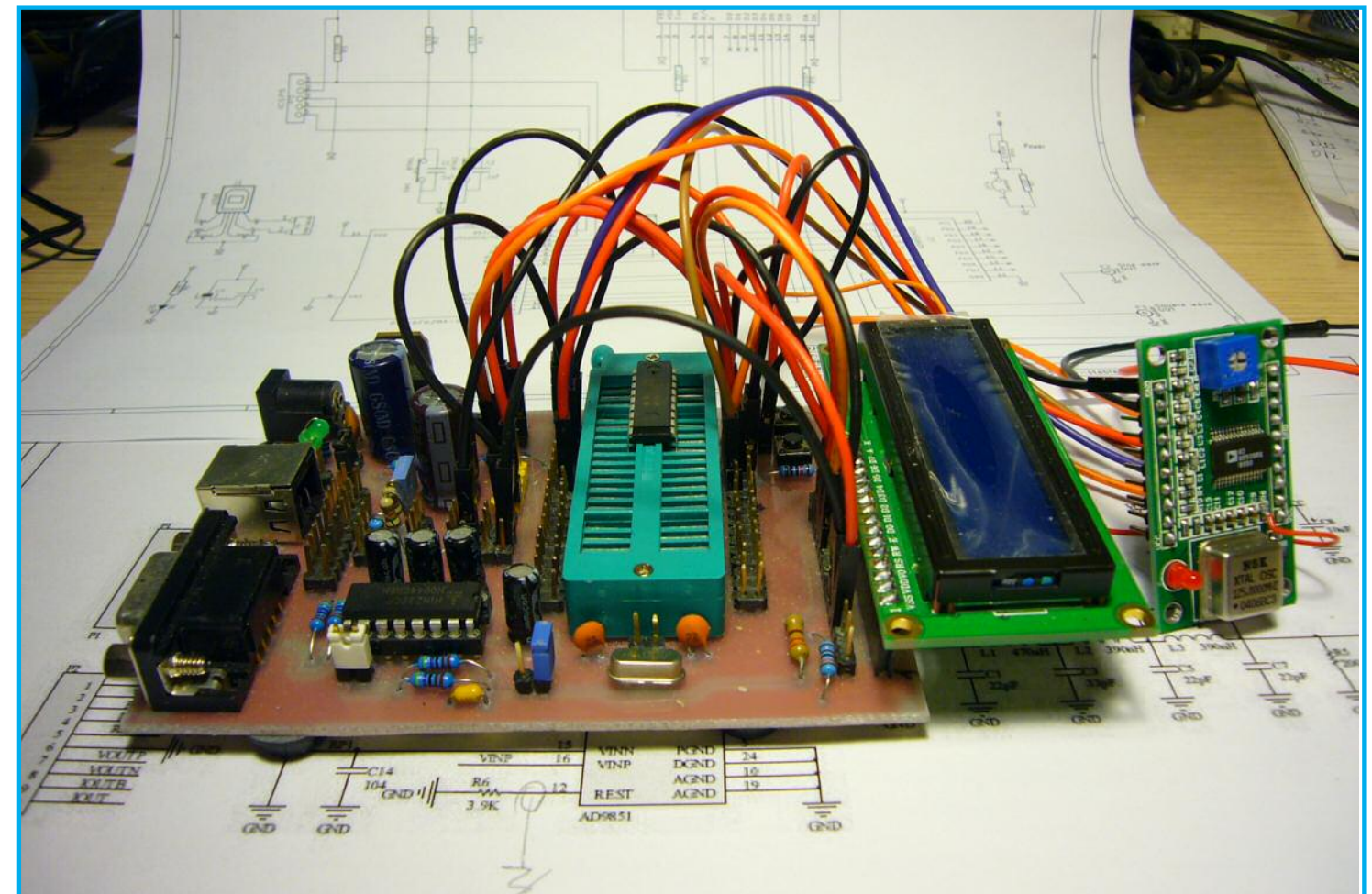



Figura 2

Figura 3

malmente un valore numerico rappresentato con due byte. Nel nostro caso la "tuning words" è rappresentata con quattro byte.
Ovviamente il modo più semplice per ottenere la frequenza richiesta consiste nel far scegliere all'operatore, attraverso un sistema di input (ad esempio pulsanti), la frequenza desiderata e calcolare la "tuning word da passare al sintetizzatore utilizzando un microcontrollore di gestione.
Per il nostro caso sono state utilizzate due piattaforme una con Arduino Uno (MCU ATMEL ATMEGA328) e l'altra con MCU Microchip PIC16F628A.
Più esattamente la piattaforma Arduino (in figura 2) è stata usata per testare velocemente, in alternativa alla classica BREADBOARD, il modulo commerciale avendo così un risultato immediato, mentre il PIC16F628A, molto più economico dell'ATMEGA, è stato usato per il progetto definito ottenendo comunque un primo prototipo sperimentale sulla mia "JREBOARD" (in figura 3).

I MODULI COMMERCIALI CON AD9850 E AD9851

Come accennato negli ultimi anni c'è stata una notevole evoluzione commerciale intorno ai circuiti AD985X e questo grazie anche alla necessità di abbinarli ad Arduino (http://www.arduino.cc). In particolare qualche azienda cinese ha sviluppato dei moduli "standard " SMD (vedi figura 4) ricavandoli dallo schema dell'applicazione descritto sul datasheet ufficiale della Analog Devices. Personalmente ho trovato solo due tipologie di schemi di montaggio e per flessibilità ho utilizzato il modulo commerciale visibile nella figura 4 e facilmente reperibile sempre sul noto sito di ASTE online. Tale circuito è ricavato dallo schema in figura 5.
Indipendentemente dal modulo utilizzato è possibile scambiare tra loro i chip AD9850 e AD9851 (compatibili pin to pin) a patto che si cambi anche il quarzo di clock.
NOTA: va sottolineato che un circuito DDS rende inutile l'uso di un frequenzimetro integrato per il controllo della frequenza di uscita (che in alcuni casi può falsare la misura) a patto che il CLOCK utilizzato anche come frequenza di riferimento, sia preciso e soprattutto stabile.
I due circuiti differiscono tra loro per la massima frequenza ottenibile:

- AD9850 max 40/45 Mhz
- AD9851 max 70/75 Mhz

In più l'AD9851 ha una serie di armoniche utilizzabili per generare frequenze più alte ma comunque non disponibili con i moduli commerciali in quanto quest'ultimi hanno normalmente un filtro passa basso tagliato a circa 70 Mhz.
Entrambi i circuiti possono essere utilizzati con un'alimentazione tra 3.3 e 5 Volts.
I moduli SMD citati sono facilmente rintracciabili sul famoso sito di aste online cercando le parole "AD9850 module" ed impostando nella ricerca avanzata la l'opzione "visualizza oggetti nel mondo".
Il costo dei moduli va da un minimo di circa 6/7 euro per l'AD9850 fino ad un massimo di 20/22 euro per l'AD9851. Ovviamente in tali prezzi indicativi gioca un ruolo non trascurabile il cambio euro/dollaro USA.

COSA POSSO FARE CON IL CIRCUITO ?

Il circuito proposto nel presente articolo, utilizzando il modulo commerciale basato su AD9850, permette di generare onde sinusoidali e quadre su frequenze da 1Hz a 45Mhz.
Modificando i codici sorgenti disponibili sul mio sito http://www.ik0jre.net nell'area download, sarà molto facile adattare il software già realizzato anche per l'AD9851 o altro circuito che utilizzi la stessa metodologia di calcolo della "tuning words" per il DDS (AD98XX). Di fatto l'utilizzatore attraverso tre tasti ed un display può impostare la frequenza desiderata agendo sui tasti stessi per modificare le singole cifre visualizzate sul display o per salvare la frequenza in uso nella memoria non volatile del PIC.
La frequenza memorizzata sarà poi disponibile al successivo utilizzo.

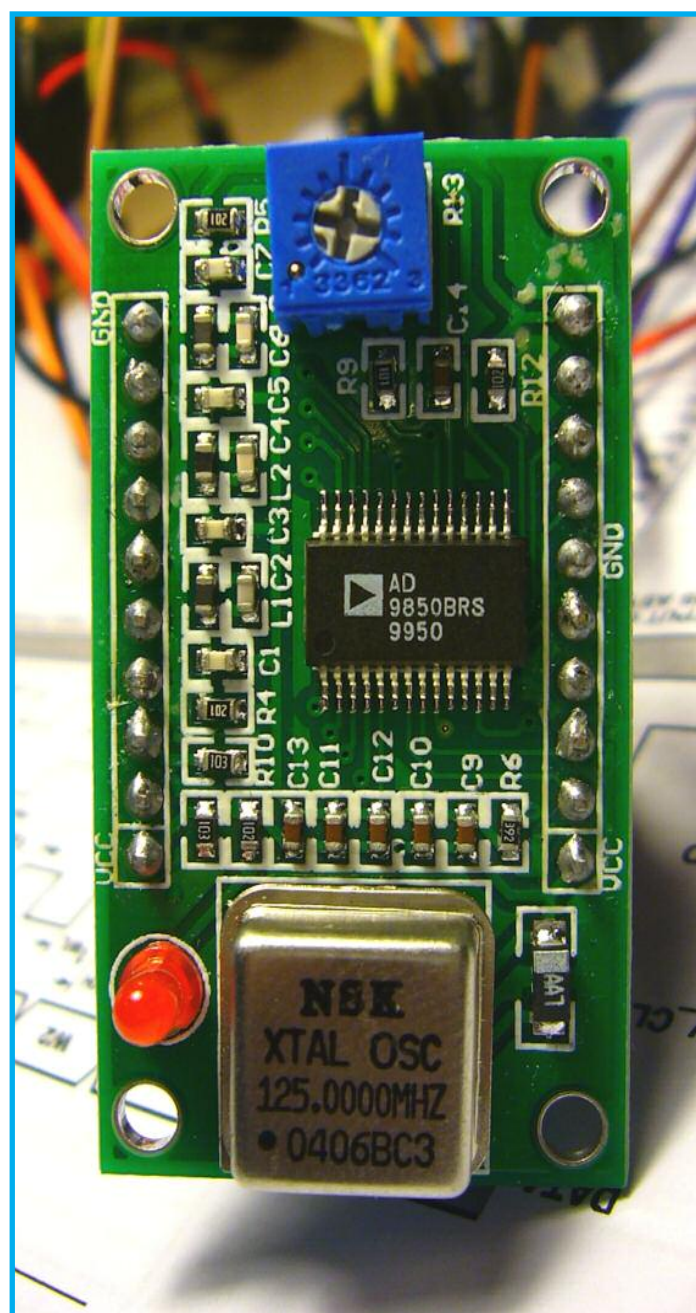
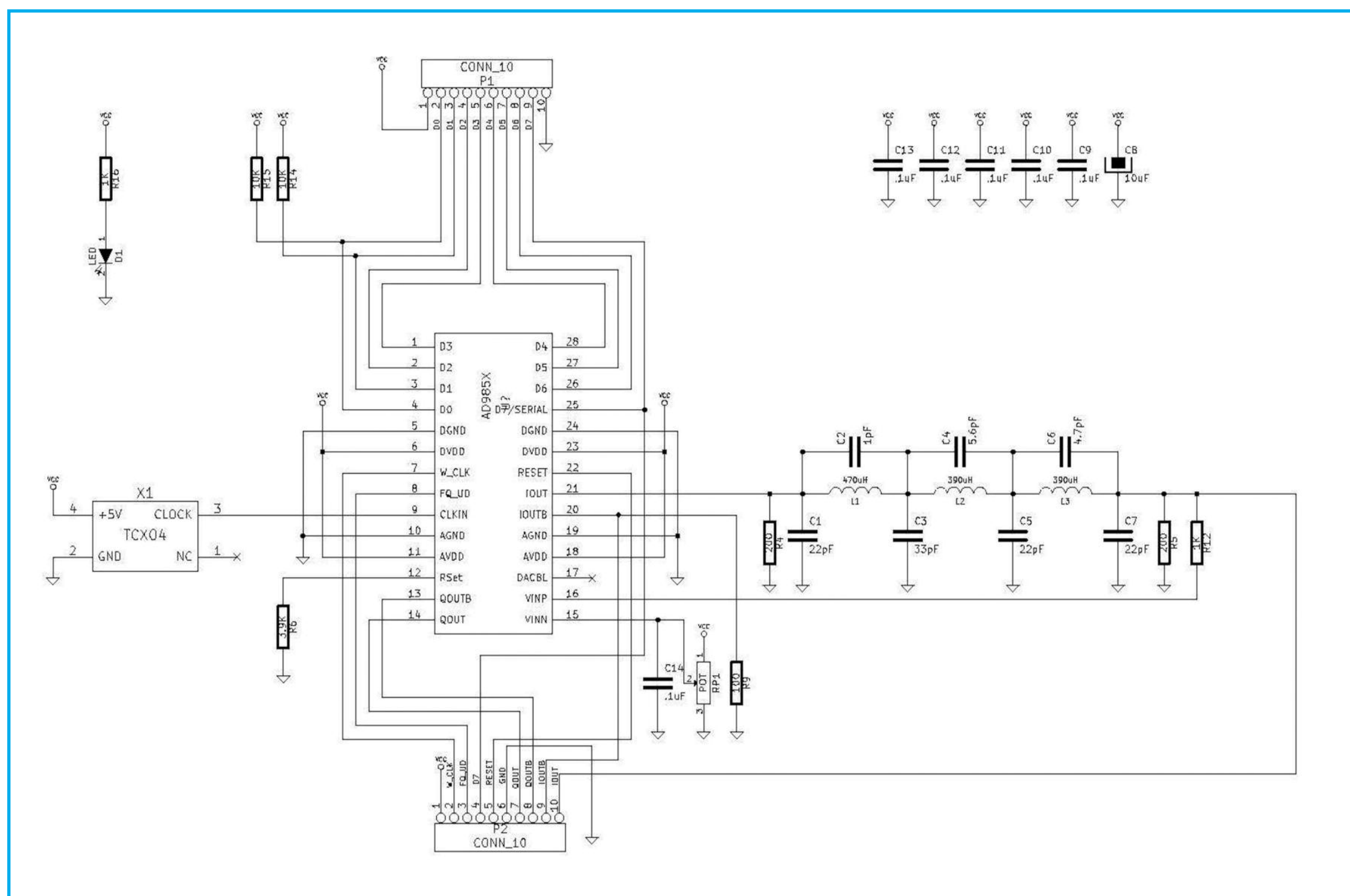



Figura 4

Figura 5

DESCRIZIONE DEL CIRCUITO

Il circuito è fondamentalmente diviso in due parti: la parte a radiofrequenza basata sul modulo DDS AD9850 e la parte logica basata su un PIC16F628A.
Avendo la parte RF già montata la rimanente parte logica diventa circuitalmente banale. Infatti, nonostante abbia già preparato un master per realizzare il circuito stampato, data la minima quantità di componenti e di connessioni in gioco, il tutto potrebbe essere assemblato su basetta millefori o addirittura sulla JREBOARD. Nella figura 6, è visibile lo schema elettrico completo del circuito.
Il circuito è alimentato da una tensione da 7 a circa 12 volts. Tale tensione, attraverso un classico regolatore 7805 fornisce i 5

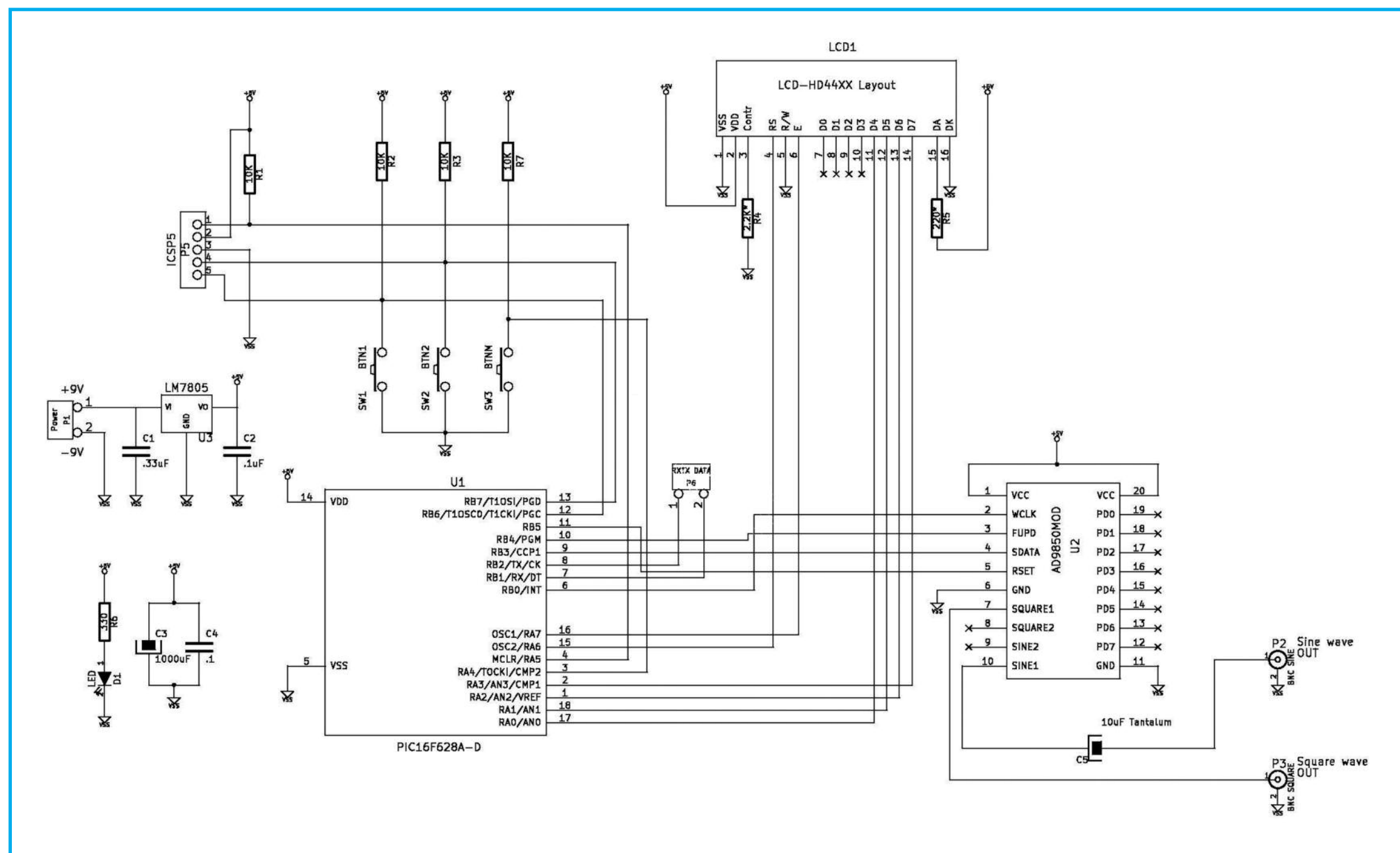



Figura 6

volts necessari all'intero circuito. Come già accennato è il "cuore del circuito" è basato sul PIC16F628A. Il PIC accetta un input dall'utente attraverso 3 pulsanti normalmente aperti e, oltre a controllare la frequenza di uscita del modulo AD, attraverso il modulo LCD 1602 (basato sul classico controller HD44780 della HITACHI ed eventuali cloni) permette di visualizzare la frequenza di uscita.
Più esattamente il modulo LCD è stato configurato nella modalità a 4 bit (linee dati D4-D7) con due linee di controllo ("Register select - RS" e "Enable – E"), collegando la rimanente linea di controllo "Read & Write – R/W" a massa abilitando così sempre la scrittura dei dati. Per risparmiare spazio il livello di contrasto è definito dalla resistenza R4 che potrebbe essere variata per alcuni display. Anche la resistenza R5 che determina la corrente massima della retroilluminazione potrebbe essere diminuita per i moduli LCD che hanno già una resistenza interna calibrata per limitare la corrente con 5 volts di alimentazione.
Per il controllo del modulo AD sono utilizzate 4 linee del PIC16F628A in quanto si è scelto di utilizzare la modalità seriale per l'invio dei dati al modulo AD ("Tuning Words" e controlli). Ovviamente la modalità seriale, alternativa a quella parallela ad 8 bit, complica un po' di più il software riducendo anche le prestazioni di diversi microsecondi ma ci permette di risparmiare 7 linee dati e visto che per il tipo di utilizzo del circuito nessuno riuscirà mai a vedere la differenza, sicuramente possiamo accettare tale configurazione.
Ipotizzando, un domani, di utilizzare la linea seriale (USART) disponibile sul PIC sono state lasciate libere le relative linee seriali del PIC stesso.
Il modulo AD fornisce due uscite separate: quella classica ad onda quadra e quella sinusoidale. Nell'uscita sinusoidale è stato inserito un condensatore di disaccoppiamento al tantalio. Va detto che per ogni forma d'onda è disponibile un ulteriore uscita, identica come ampiezza ma con una fase diversa. L'uscita ad onda quadra, destinata soprattutto a circuiti digitali non necessità di circuiti di disaccoppiamento. L'uscita a 100Khz come visibile dalle foto 7 e 8 è di circa 1 volt picco picco. Ovviamente salendo di frequenza diminuirà sensibilmente il livello di uscita.
Per completezza va detto che il circuito AD98X consente le alimentazioni separate per la parte analogica e la parte digitale ma il modulo commerciale o meglio i moduli che ho trovato, uniscono di fatto le due alimentazioni.
Con i tre pulsanti in dotazione è possibile impostare con facilità la frequenza desiderata e salvare quest'ultima nella eeprom del PIC. Per scelte costruttive il range impostabile va da 0 a 49999999 Hz. Ovviamente alle frequenze più alte il modulo AD risente notevolmente dell'alta frequenza con una notevole attenuazione e generazione di spurie sul segnale d'uscita.
Inserendo nel circuito un modulo AD9851 con una piccola modifica al software si potrebbero raggiungere i 70Mhz.
Il software può essere caricato, senza to-

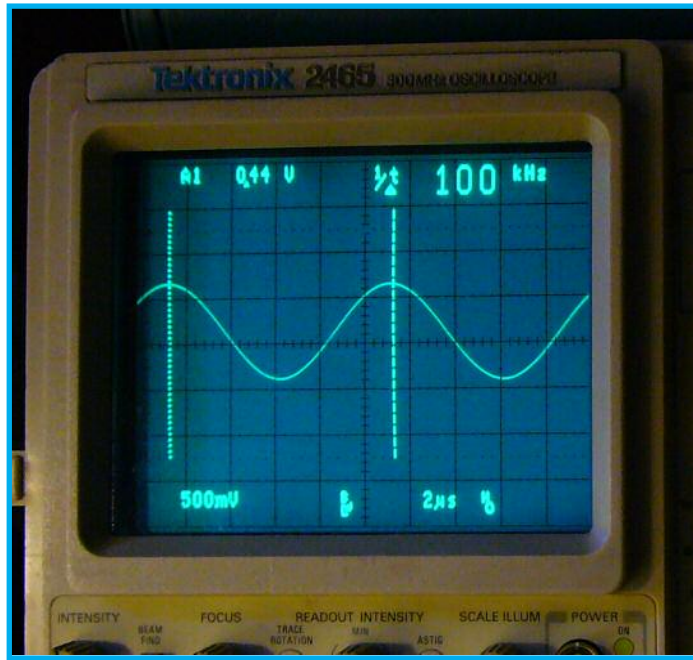



Figura 7

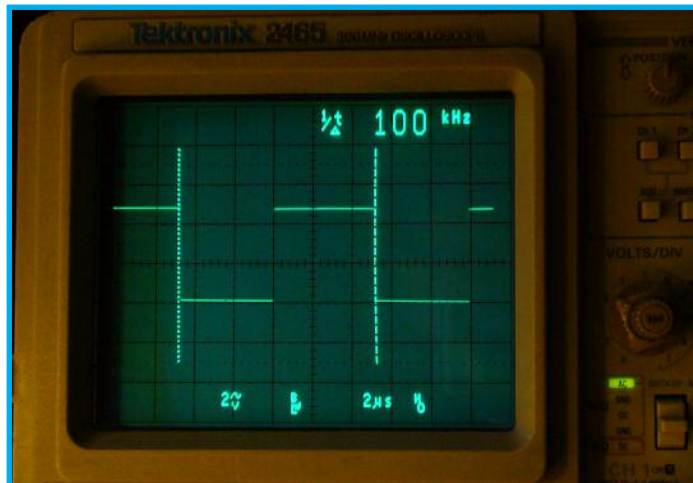



Figura 8

gliere il PIC dal circuito tramite, utilizzando un programmatore pickit2 o pickit3 (e relativi cloni) connesso al connettore ICSP P5.

COSTRUZIONE DEL CIRCUITO

Avendo a disposizione il modulo commerciale basato sull'AD, come già accennato, tutto il resto potrebbe tranquillamente essere montato su basetta millefori avendo cura di mantenere molto corti le 2 linee di uscita del segnale.

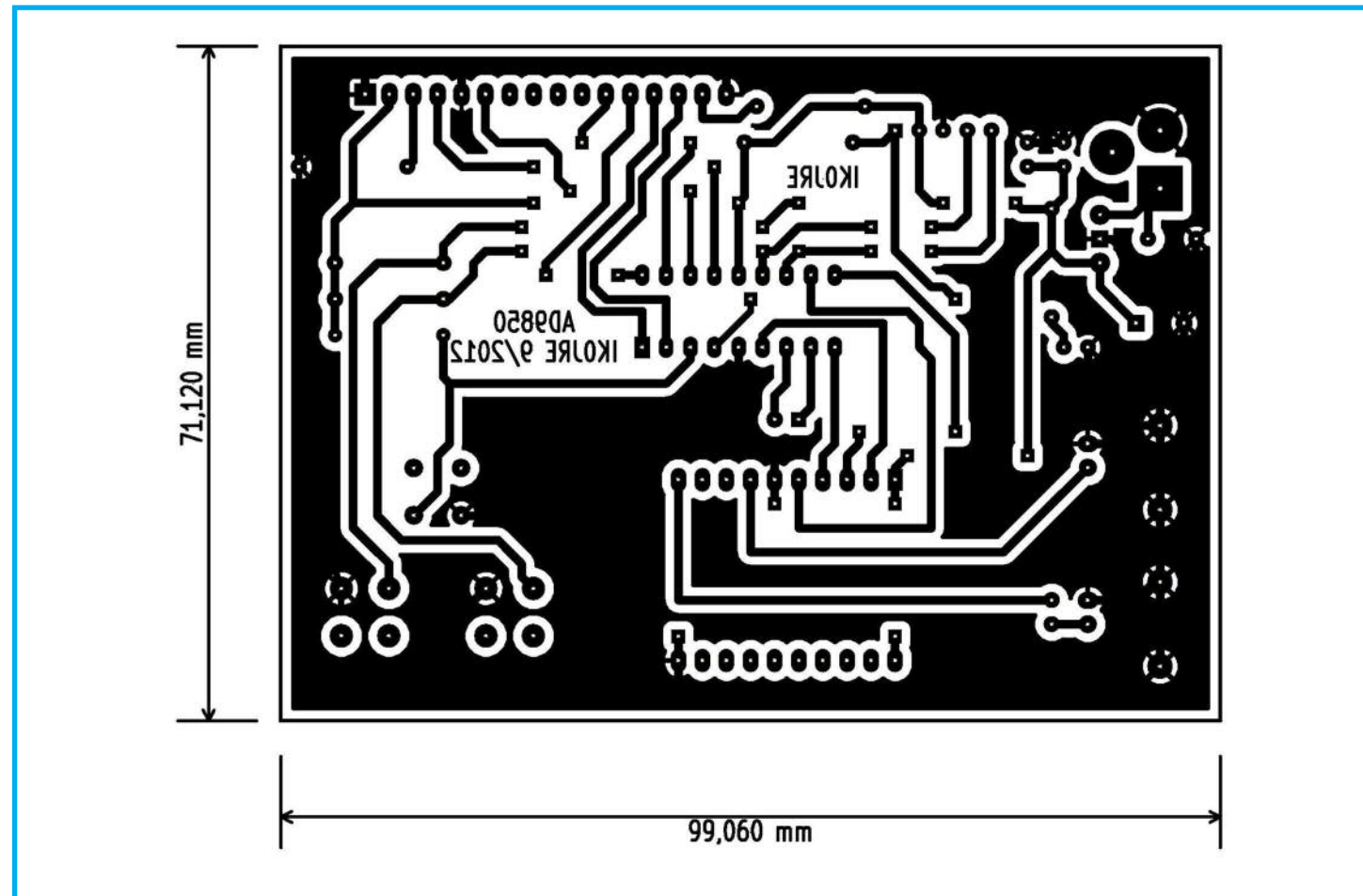



Figura 9

LISTA COMPONENTI

Condensatori
C1 .33uF, C2 C4 .1uF, C3 1000uF 16V, C5 10uF 10V tantalio;

Resistenze da ¼ di watt
R1 R2 R3 10K, R4 2.2K, R5 220, R6 330, R7 10K;

Componenti attivi
U1 PIC16F628A (con zoccolo 18 pin), U2 modulo commerciale AD9850, U3 LM7805;

Connettori sil
P5 SIL 5pin, P6 SIL 2pin;

Varie
D1 diodo led 3mm,
LCD1 è un modulo basato su chip HD44XX tipo 1602,
SW1, SW2, SW3 sono pulsanti normalmente aperti con passo da 5mm,
P1 è un connettore alimentazione a spina coassiale da circuito stampato,
P2 e P3 sono due connettori BNC da circuito stampato recuperati da vecchie schede ethernet per bus ISA.

Comunque, per la produzione di un circuito stampato, nelle seguenti figure 9 10, vediamo sia il master (a specchio) del circuito stampato sia la disposizione dei componenti dell'ultima versione da me prodotta (la maggior parte delle foto invece si riferisce al primo prototipo nel quale per comodità non era stato usato un regolatore di tensione ma una presa USB di tipo B).
Per il modulo LCD e il modulo AD sono usati 3 connettori classici di tipo SIL femmina da 9mm di altezza, rispettivament uno16 pin per LCD e due da 10 pin per il modulo AD.
Il modulo AD viene venduto già con i connettori SIL maschi mentre il modulo LCD è spesso venduto senza connettore e quindi necessità la saldatura di quest'ultimo.
Rispettando le altezze dei connettori il display si sovrappone alla maggior parte dell'elettronica mentre il modulo AD si sovrappone parzialmente al condensatore C5.
Il connettore P5, utilizzato per la programmazione ICSP del PIC, di tipo SIL 5pin maschio, è indispensabile che sia del tipo a 90° (foto 11 a lato) per poterlo utilizzare senza smontare il display.
Essendo il circuito stampato monofaccia, per il collegamento tra alcune piste, è necessario realizzare i 16 ponticelli segnati in rosso nella disposizione dei componenti della foto 10.
Nella foto 12 si può vedere il dettaglio laterale del display con dei distanziatori montati onde evitare che con il tempo il connettore sul circuito si deformi.
Il montaggio dovrebbe proseguire sulla base delle altezze dei componenti.

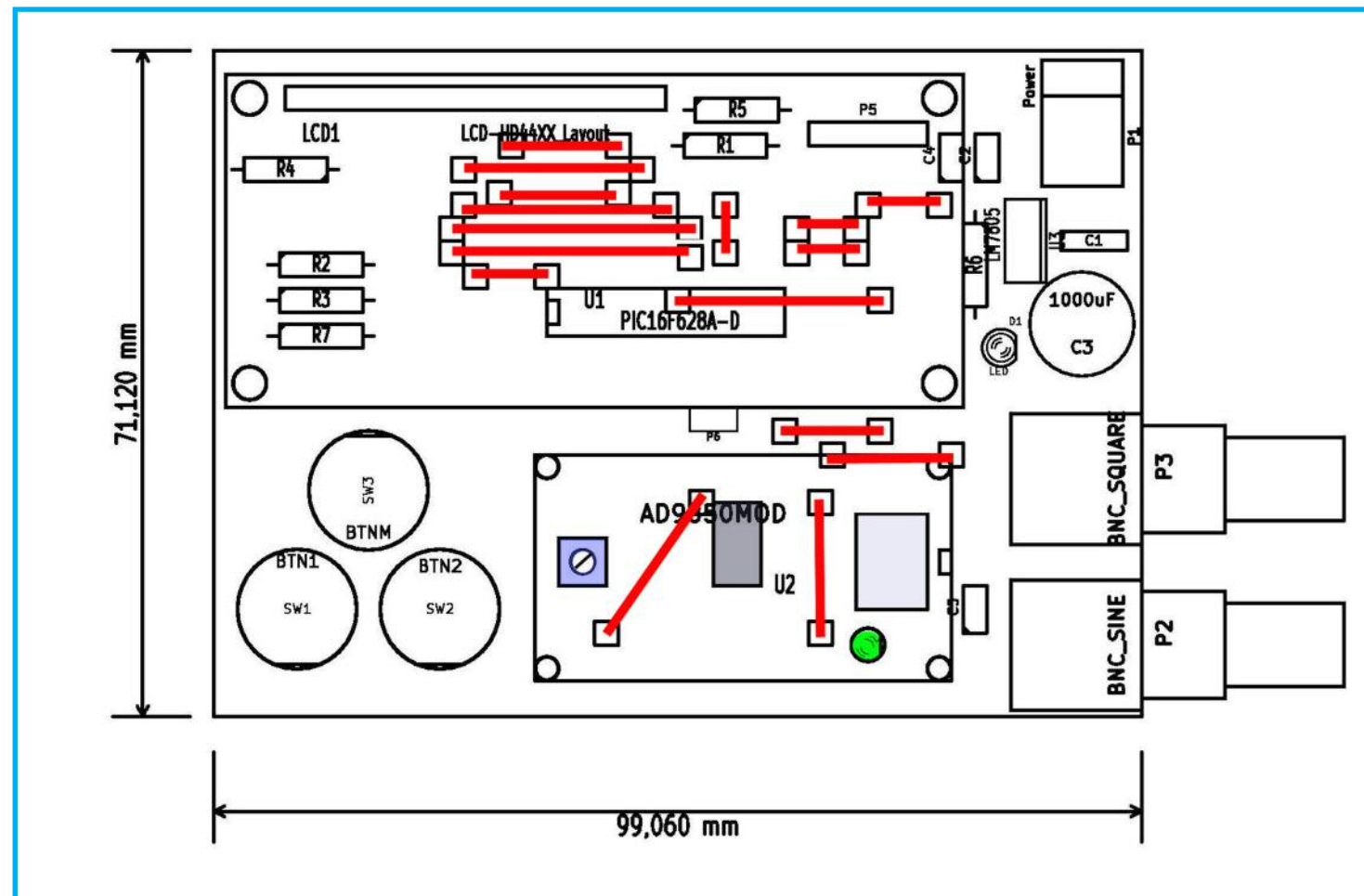



Figura 10

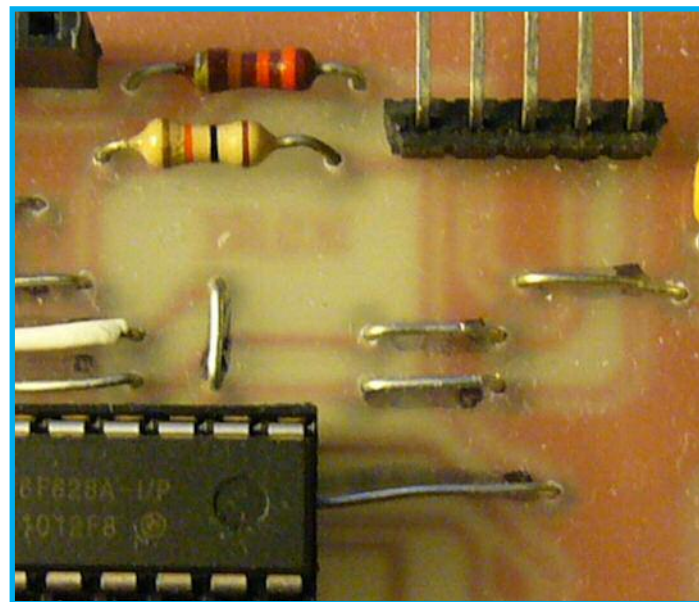



Figura 11

Terminato il montaggio prima di inserire i 3 componenti attivi su zoccolo (PIC16F628A, LCD e modulo AD) vi consiglio di controllare la presenza dei 5 volt per i moduli in questione. Fate anche attenzione alla corretta direzione nell'inserimento del modulo AD onde evitarne una morte prematura (come riferimento il diodo led verso il connettore BNC ed il trimmer verso i pulsanti). Per completezza trovate anche i layout degli altri componenti attivi (foto 13 e 14).
Montato il tutto se avete un programmatore seriale per PIC potete scaricare il software in formato .HEX dall'area download del mio sito (http://www.ik0jre.net) e caricarlo prima di inserire il PIC nel circuito. In questo modo il circuito sarà immediatamente funzionante all'accensione.
In alternativa potrete completare il circuito e programmare il PIC direttamente sul circuito attraverso il connettore ICSP (In Circuit Serial Programming). In quest'ultimo caso il display sarà acceso ma non mostrerà alcunché fino ad avvenuta programmazione.

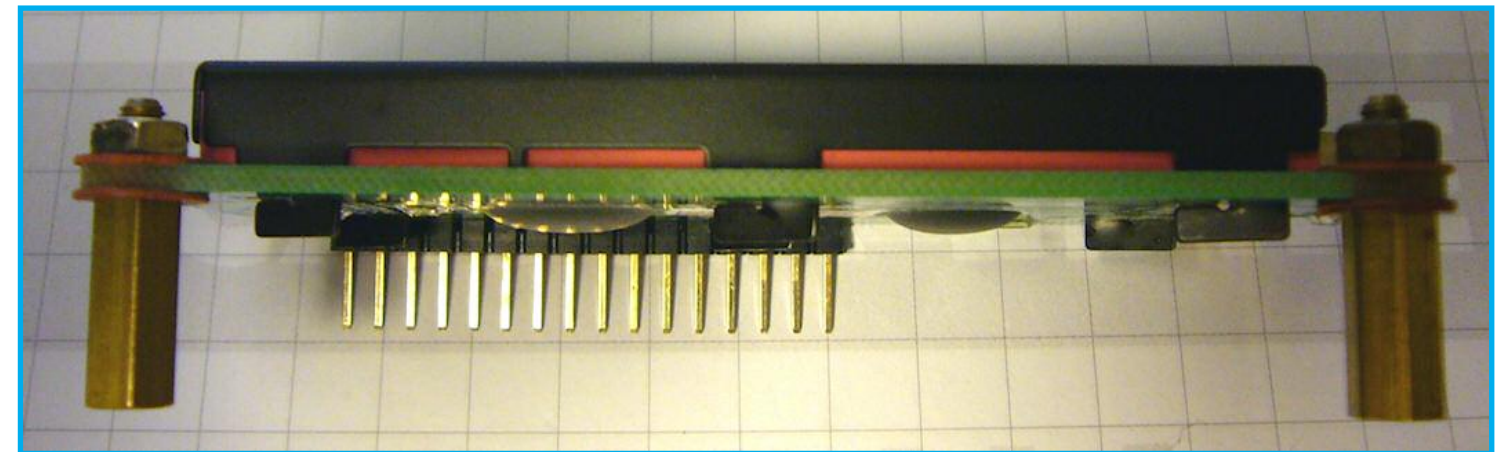



Figura 12

Anche il codice sorgente del software è disponibile nell'area download del mio sito. Come sempre il software è utilizzabile e modificabile gratuitamente, anche per scopi commerciali, con l'unica condizione di citare l'autore e la provenienza.

SOFTWARE ED UTILIZZO DEL GENERATORE

Purtroppo non c'è spazio per descrivere le modalità di analisi e scrittura del codice sorgente perché abbastanza complesso. Complessità dovuta anche alla carenza nei PIC della serie 16, di funzioni dedicate alle operazioni di divisione e moltiplicazione in virgola mobile; per tale motivo credo quindi sia più utile descriverne in modo semplice il solo utilizzo utente e lasciando allo sperimentatore lo studio del codice sorgente tra l'altro molto commentato.
L'unico intervento da parte dell'utente è l'impostazione della frequenza di lavoro. Tale impostazione della frequenza può essere fatta attraverso i tre pulsanti a disposizione (vedi figura 16). Più esattamente i due pulsanti in basso operano sulla frequenza (incrementandola con quello di sinistra e decrementandola con quello di destra) mentre il terzo pulsante aggiunge un'ulteriore funzionalità ai primi due come si vedrà poi.
Per capire come agiscono i pulsanti si deve dare un'occhiata al display: all'accensione il display (figura 15) mostra, in Mhz, nella linea in basso, l'ultima frequenza memorizzata.
La linea superiore, che possiamo considerare come "linea di status", mostra sulla destra un indicatore di selezione, mentre sulla sinistra la selezione stessa.
L'indicatore di selezione non è altro che l'indicatore di quante unità verrà incrementata o decrementata la frequenza. O se vogliamo può essere definito come l'indicatore dello STEP di incremento e decremento.
Tenendo premuto il tasto centrale (function) mentre si tiene premuto anche il tasto di destra (down) si sposta il selettore di incremento/decremento per impostare l'opportuno step di frequenza. Il selettore si sposterà circolarmente da destra verso sinistra e contemporaneamente cambierà anche l'indicazione dello step attivo a sinistra. Arrivato al massimo step possibile tornerà al suo minimo di sinistra. Gli step possibili sono 8 partendo da 1Hz si arriva ai 10Mhz (moltiplicando per 10 ad ogni step successivo). I pulsanti Up e Down in-

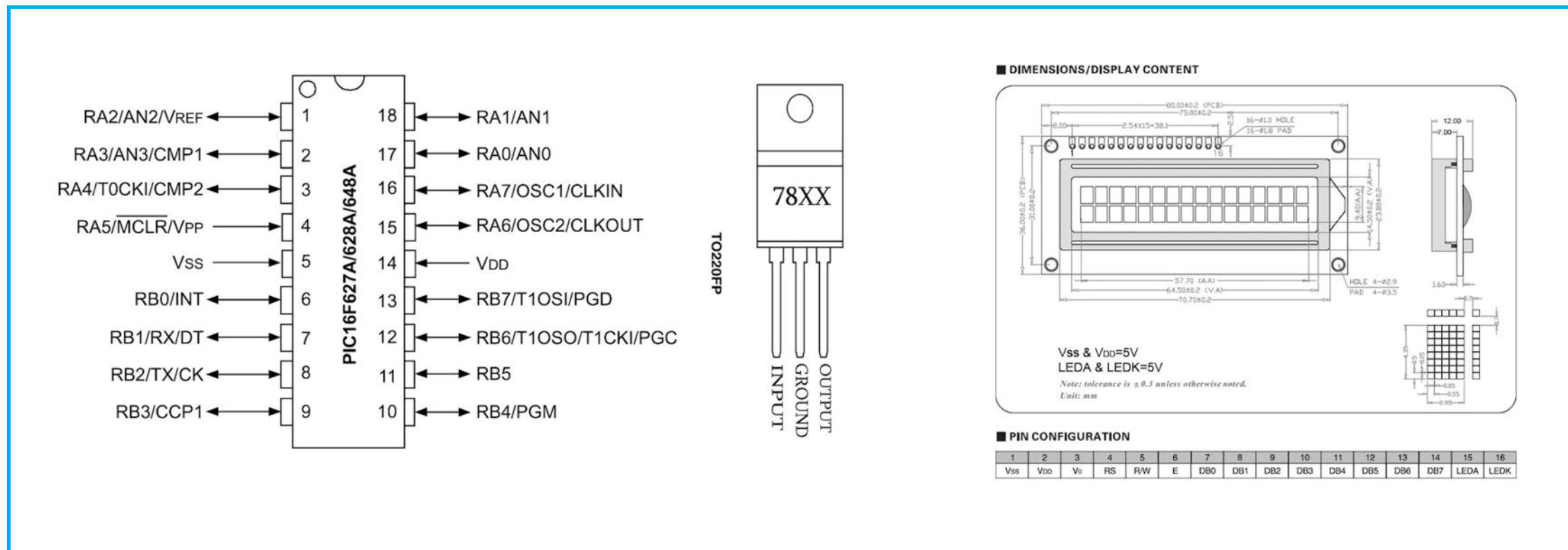



Figura 13

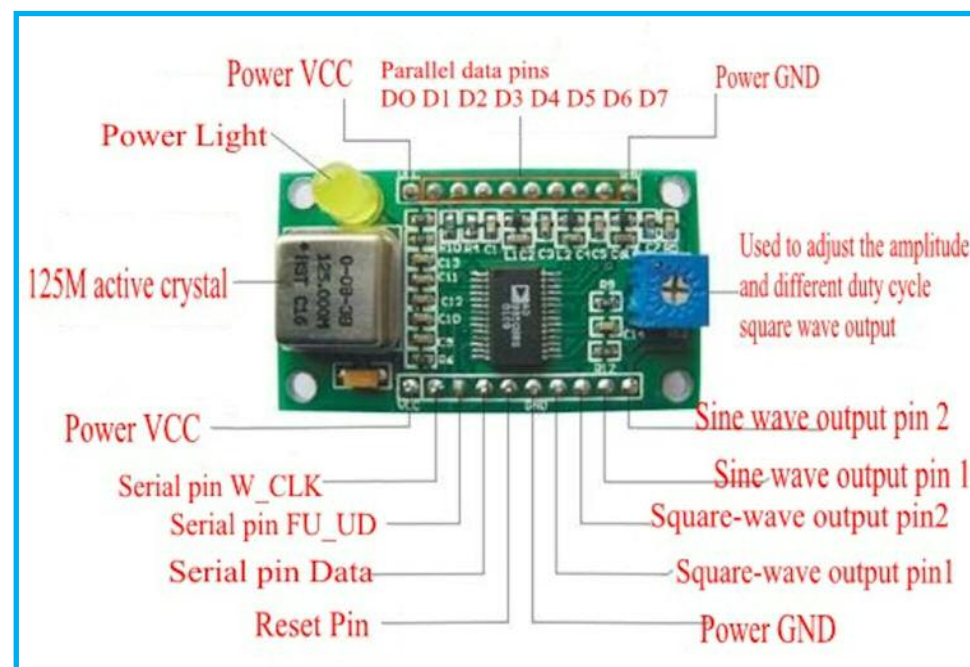



Figura 15

Figura 14

crementano e decrementano rispettivamente la frequenza di una unità di step se premuti lentamente mentre mantenendo il pulsante premuto per oltre 2 secondi circa, l'incremento sale a 5 unità di step.
Per fare un esempio: se si volesse impostare una frequenza di 1150000 Hz (1 MHz e 150 KHz),
1) ci sposteremo con il selettore sopra i la quarta cifra da sinistra, con l'indicatore di step indicante i 10KHz dopodiché si premerà il pulsante "up" fino a visualizzare il numero 5;
2) ci sposteremo con il selettore sopra i la terza cifra da sinistra, con l'indicatore di step indicante i 100KHz dopodiché si premerà il pulsante "up" fino a visualizzare il numero 1;
3) ci sposteremo con il selettore sopra i la seconda cifra da sinistra, con l'indicatore di step indicante i 1MHz dopodiché si premerà il pulsante "up" fino a visualizzare il numero 1. L'impostazione delle singole decadi è una scorciatoia per l'impostazione della frequenza che comunque può essere fatta come un qualsiasi VFO: ad esempio avendo una frequenza visualizzata di 1950000 (1MHz e 950KHz) e con con il selettore sopra i 100KHz, premendo il pulsante "UP", la nuova frequenza diventerà 2050000 Hz (2 Mhz e 50KHz).
Volendo salvare la frequenza impostata, tenendo premuto il tasto centrale (function) mentre si tiene premuto anche il tasto di sinistra, si trasferirà la frequenza attualmente attiva nella memoria non volatile del PIC (eeprom) per averla immediatamente disponibile alla successiva accensione.
Nonostante la memoria programma del PIC sia utilizzata solo per metà, per il momento non sono previsti ulteriori funzioni anche se è in "cantiere" un'implementazione per aggiungere la generazione della frequenza a spazzolamento (sweep).

NOTE E CONCLUSIONI

Credo sinceramente che per alcuni di voi il progetto può essere un ottimo punto di partenza per sviluppare il proprio software in quanto utilizzando alcune tecniche presenti nel codice sorgente (che mi hanno impegnato per diversi giorni di analisi) si semplificano notevolmente molte operazioni (alternative alla virgola mobile) rimanendo entro dei ragionevoli limiti di precisione.
Un'altra implementazione utile potrebbe riguardare un adattatore d'impedenza comprensivo di circuito attenuatore di precisione per la calibrazione di apparecchiature o taratura filtri.
Sapendo comunque di non essere stato completamente esaustivo concludo augurandovi un buon divertimento e vi invito a contattarmi per eventuali chiarimenti in merito al presente progetto.

ALTRE INFORMAZIONI

Sia l'hardware, sia il software sono stati interamente realizzati dal sottoscritto; il software è scaricabile gratuitamente dal sito web http://www.ik0jre.net nell'apposita sezione dedicata ai PIC. Tutto quanto dispo-

nibile sul sito web può essere utilizzato e modificato liberamente per scopi sia privati sia commerciali con l'unica condizione, per entrambi i casi, di citare l'autore in ogni copia riprodotta e/o modificata e sull'eventuale documentazione a corredo.
Ulteriori informazioni riguardo ai PIC sono reperibili sul sito del produttore Microchip (www.microchip.com), titolare del marchio registrato PICMicro, MPLAB e là dove non espressamente citato. La documentazione Microchip è la bibliografia di riferimento per tutta la documentazione riguardante i PIC e la loro programmazione.
HD44780 è un prodotto della HITACHI, produttrice di buona parte nella documentazione di riferimento attualmente in circolazione.I circuiti AD9850 AD9851 sono prodotti dalla società Analog Devices (http://www.analog.com) proprietaria dei relativi copyright associati.
I circuiti stampati sono stati realizzati con basetta FR4 presensibilizzata esposta ai raggi UV per 40 secondi con il bromografo artigianale.
Vi ricordo inoltre che pur avendo posto la massima attenzione nello scrivere il presente articolo non sono esclusi eventuali errori od omissioni e per questo motivo ne il sottoscritto ne l'editore possono essere ritenuti responsabili direttamente o indirettamente di eventuali danni arrecati a cose o persone nella realizzazione di quanto esposto nel presente documento.

Figura 16

a cura della redazione

MICROCONTROLLORI

PILOTARE CARICHI DI POTENZA

"Smanettando" con i microcontrollori vi sarà sicuramente capitato di dover interfacciare un carico di potenza alle uscite del micro. Ecco alcune tecniche utili allo scopo

Solitamente un microcontrollore non è in grado di pilotare carichi di potenza. Esso infatti può gestire piccoli dispositivi con assorbimento di corrente pari a pochi milliampere. Se una MCU deve illuminare un diodo Led o pilotare un display LCD, il problema non sussiste.
Ma cosa succede invece quando il micro deve comandare l'accensione di un motore o di una lampada di potenza? L'articolo passa in rassegna alcuni metodi di collegamento inerenti le uscite di un microcontrollore. Qualunque tipologia di microcontrollore può essere utilmente adoperata, rispettando le tematiche teoriche trattate.

COLLEGAMENTO DIRETTO DI UNA SOLA USCITA

Si tratta del collegamento tradizionale e quindi più semplice. Esso prevede la connessione diretta tra una porta d'uscita della MCU e il carico da attivare. Dal momento che, solitamente, ogni porta è capace di erogare pochi milliampere di corrente (circa 25 mA), possono essere collegati solamente piccoli carichi, come diodi Led, canali logici di un display LCD, oppure gli ingressi logici di ulteriori apparecchiature collegate a valle. In questi casi non bisogna mai eccedere la massima corrente sopportata dalla porta stessa poiché, oltre ad ottenere uno scompenso di corrente dell'intero circuito e un riscaldamento anomalo dell'integrato (in tali casi pericoloso), si potrebbe incorrere in un calo di prestazioni dello stesso, con una erogazione "più

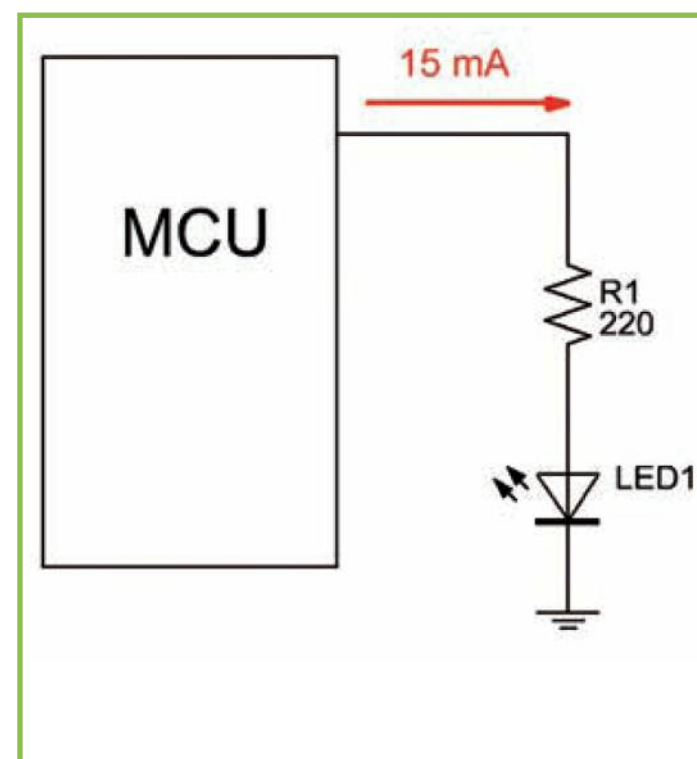



Figura 1: una porta di MCU che pilota direttamente un carico

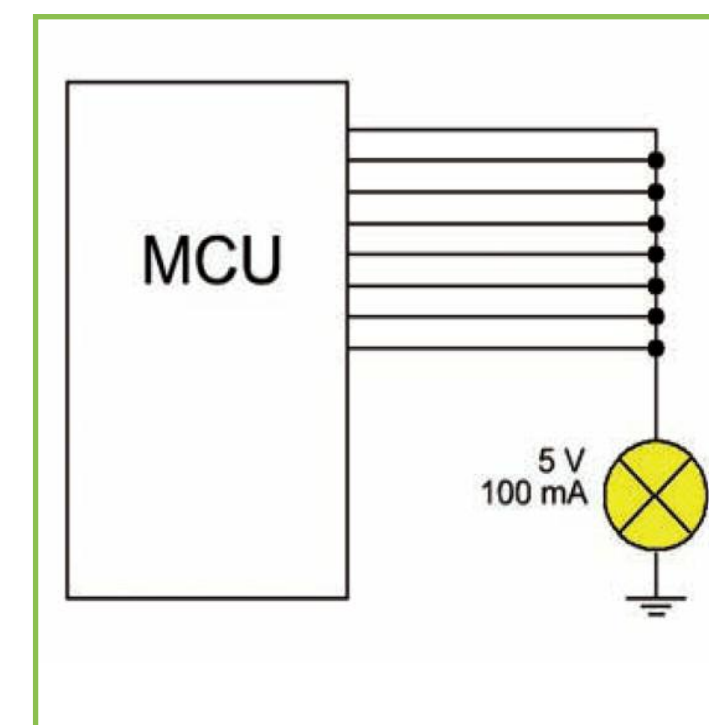



Figura 2: otto porte di una MCU che pilotano direttamente una lampada da 100 mA.

lenta” verso la porta, per la presenza di “rampe in salita” indesiderate, soprattutto alle alte velocità. Ci si limiti dunque a utilizzare valori di corrente veramente bassi, molto al di sotto della soglia sopportata teorica.

COLLEGAMENTO DIRETTO DI PIÙ USCITE

Nel caso in cui è richiesta una più alta aliquota di corrente, e non si vogliono utilizzare soluzioni esterne più complicate, è possibile “farsi aiutare” da uscite supplementari dello stesso micro, al fine di moltiplicare la quantità di corrente erogata. In questo caso è possibile usare il “parallelismo” di più porte, sfruttando la somma algebrica di corrente. In questo caso occorre però seguire le seguenti regole:

- le porte coinvolte devono essere attivate o disattivate nello stesso istante. Dal punto di vista elettronico, in realtà, esse avranno un momento di attivazione successivo pari al momento di latenza delle singole istruzioni di programma adibite a tale compito, ma ai fini pratici tale ritardo di tempo è da considerarsi ininfluente;
- anche se il carico applicato riesce ad essere alimentato dalla corrente massima erogata da ogni porta, non bisogna mai superare quella massima erogata dal microcontrollore per la totalità delle porte. Questo specifico caso ci permette di eseguire una prova sul famoso ma vetusto 16F84: come detto prima, la corrente massima di ogni porta ammonta a circa 25 mA. Se il progettista si basasse erroneamente solo su tale parametro, egli potrebbe collegare otto diodi led sulle otto porte, per un assorbimento totale di 8 x 25 mA = 200 mA.

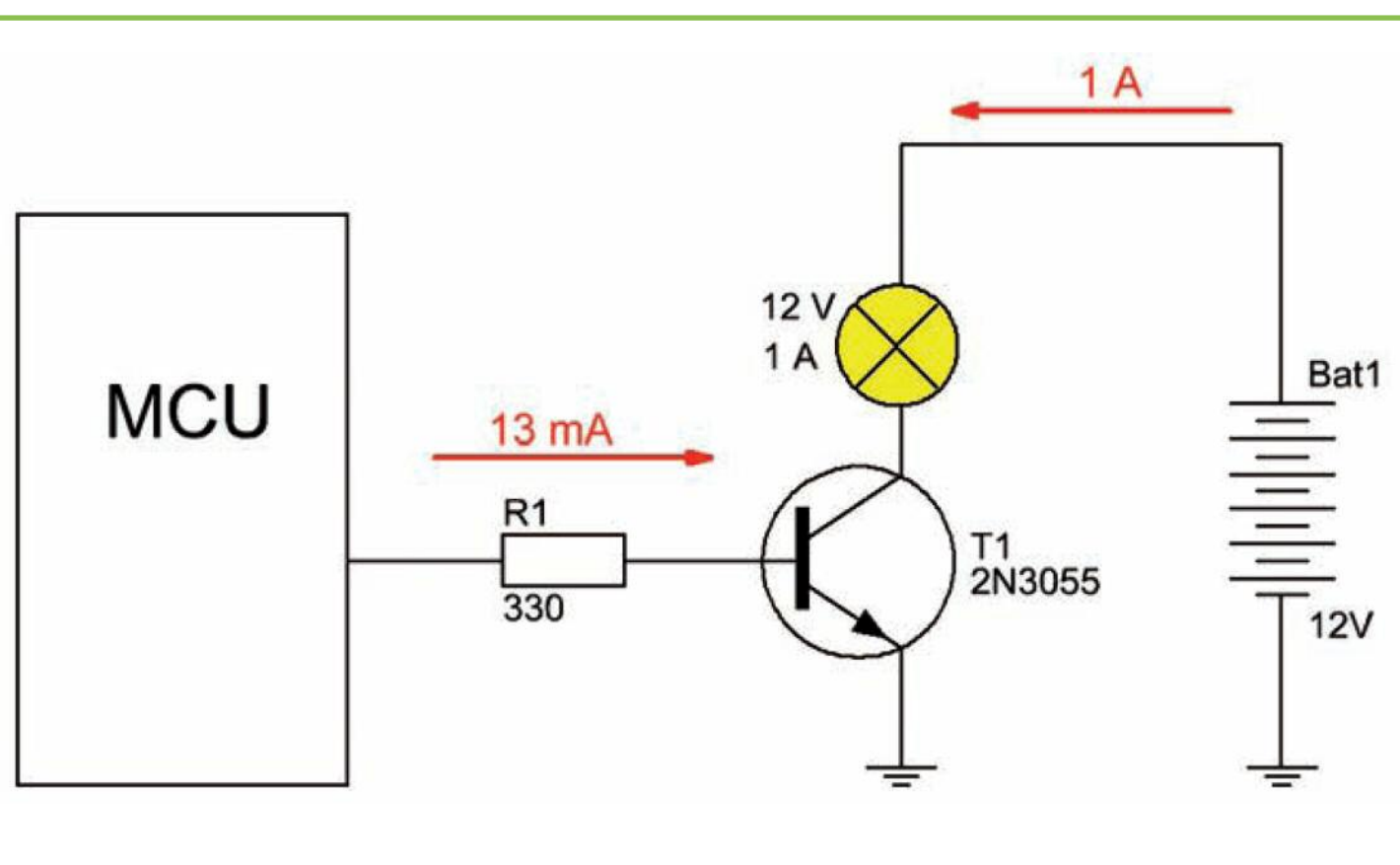



Figura 3: un microcontrollore che pilota una lampada ad incandescenza di 1 A tramite un transistor.

Purtroppo l’integrato riesce a sopportare al massimo 150 mA. Occorre dunque considerare anche la massima circolazione consentita sul pin VSS, che è poi quello che dà corrente all’intero circuito logico del micro. Utilizzando questo metodo è possibile illuminare, ad esempio, una piccola lampadina ad incandescenza, con il coinvolgimento dei seguenti valori elettrici, secondo lo schema visualizzato in **figura 2**:

- lampada funzionante con una tensione di 5 V;
- corrente richiesta dalla stessa di 100 mA (superiore quindi a quella erogabile da un singolo piedino);
- corrente massima erogabile da una singola porta: 25 mA;
- corrente massima teorica erogabile da 8 porte: 8 x 25 mA = 200 mA;
- corrente massima reale erogabile da 8 porte: 150 mA;
- corrente erogata da ogni porta, con il nostro esempio: 100 mA : 8 = 12,5 mA. Come si vede dall’esempio, il sistema è in grado dunque di illuminare una lampada di 100 mA, sfruttando otto porte che erogano solamente 12,5 mA.

L’unico svantaggio del metodo è quello di sacrificare tante porte, altrimenti utilizzabili per altri scopi. Si ricorra ad esso quando il numero di uscite della MCU è sufficientemente grande.

UTILIZZO DEL TRANSISTOR

Quando la corrente richiesta comincia ad aumentare, le porte del micro, da sole, non hanno sufficiente potenza per pilotare i carichi esterni. Ragion per cui è indispensabile richiedere un piccolo aiuto ad altri componenti elettronici. E’ il caso del transistor, di potenza o non, che con la sua elevata corrente di collettore e la sua bassa corrente di base, riesce egregiamente a risolvere il problema. Occorre in questi casi rispettare alcune condizioni:

- la corrente della porta di uscita della MCU deve essere sufficiente a pilotare la base del transistor, e questo è un fatto sicuro, dal momento che, solitamente, la base assorbe pochi microampere di corrente per la maggior parte dei modelli;
- la tensione di alimentazione deve poter erogare una corrente adeguata al carico presente sul collettore;
- la frequenza di taglio del transistor deve rispondere alla frequenza di pilotaggio dettata dal microcontrollore;
- la saturazione del transistor, con conseguente passaggio di corrente, non deve causare un drastico abbassamento della tensione di alimentazione della MCU;
- nel caso di correnti elevate è opportuno dotare il transistor di adeguata aletta di raffreddamento. L’adozione del metodo richiede, da parte del progettista, un’attenta valutazione soprattutto sul dimensionamento delle resistenze di polarizzazione del componente. L’esempio che segue mostra come un microcontrollore possa pilotare una lampada ad incandescenza che assorba 1 A di corrente e alimentata a 12 V. Ovviamente la corrente è prelevata da un generatore esterno, in quanto il microcontrollore funge solo da “attivatore” del transistor, quindi da elemento intelligente, mentre il massimo carico di lavoro è affidato al noto 2N3055. Per potenze superiori, la base potrebbe

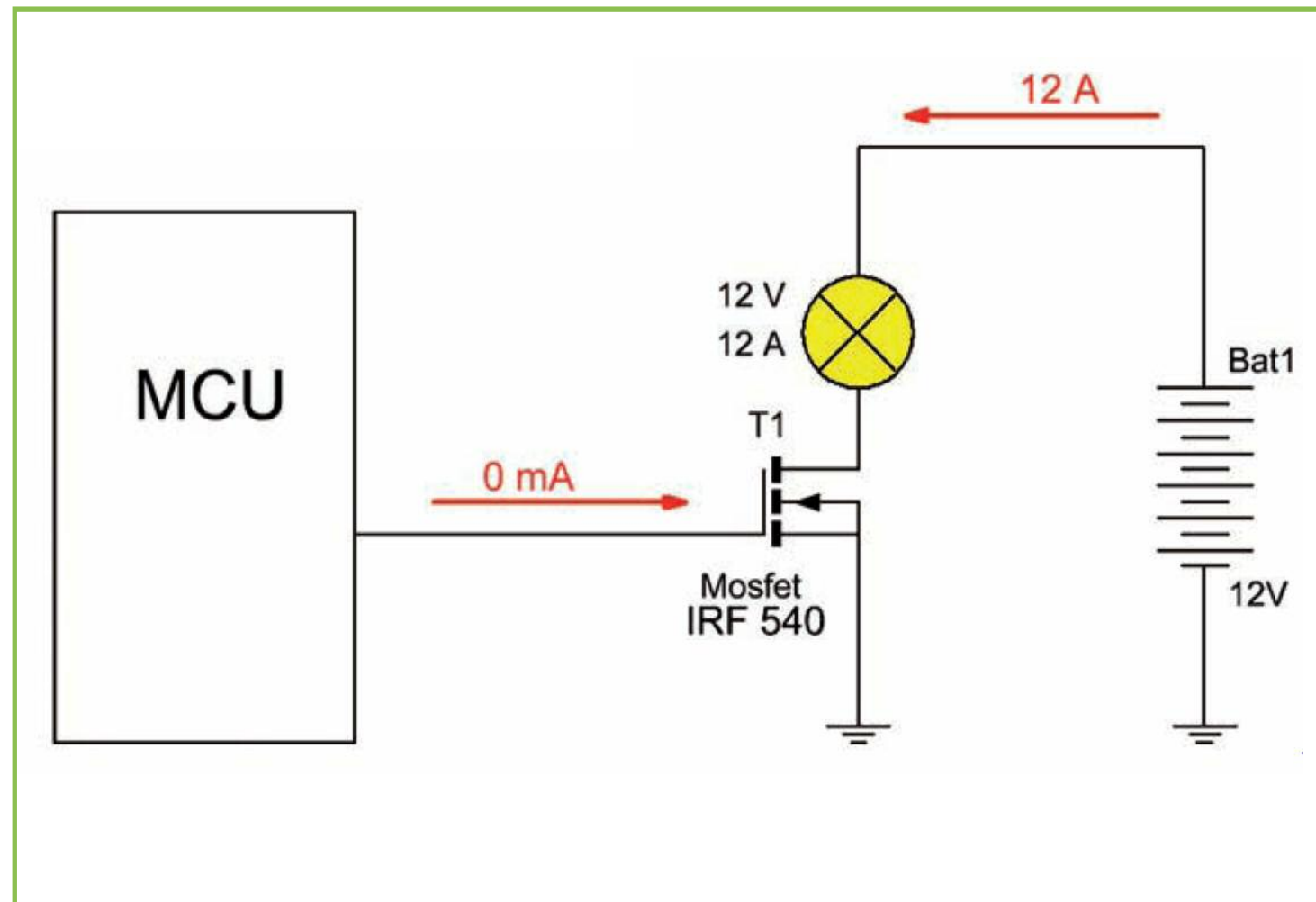



Figura 4: un microcontrollore pilota una lampada ad incandescenza di ben 12 A tramite un Mosfet

richiedere una erogazione di corrente maggiore, rispetto a quella disponibile per ogni porta di uscita. In questi casi è sufficiente utilizzare due o più pin, come abbiamo visto nel precedente esempio.

UTILIZZO DEL MOSFET

Utilizzare un Mosfet per pilotare grossi carichi consente di ottenere molti vantaggi:

- permette di ridurre a zero la dissipazione elettrica e termica, dal momento che la VDS~0 e la Rds<1/10 ohm;
- permette di ottenere una ottima velocità di commutazione, specialmente in applicazioni a PWM;
- permette di utilizzare correnti molto elevate;
- consente un dimensionamento più agevole dei componenti.

L'esempio che segue prevede una erogazione molto alta di corrente. Un microcontrollore riesce infatti a pilotare una lampada ad incandescenza di 12 A con tensione di 12 V, in pratica una potenza assorbita di ben 144 Watt. Grazie alla potenza sopportata dal Mosfet e, soprattutto, all'assenza di corrente di pilotaggio del Gate (i Mosfet infatti si pilotano in tensione), l'implementazione del circuito è abbastanza semplice.

Occorre prestare attenzione al fatto che la corrente, quando supera i 5 A, crea non pochi problemi nella gestione del PCB, per cui bisogna prestare attenzione agli spessori delle piste su circuito stampato e alle sezioni dei cavi utilizzati.

Con questo metodo, la porta di uscita del micro non viene assolutamente "stressata", in quanto basta la sola tensione del livello logico alto per innescare il Mosfet. Non è quindi necessaria alcuna resistenza di base di limitazione.

Figura 5: un microcontrollore pilota un carico resistivo di ben 60 A utilizzando tanti Mosfet in parallelo

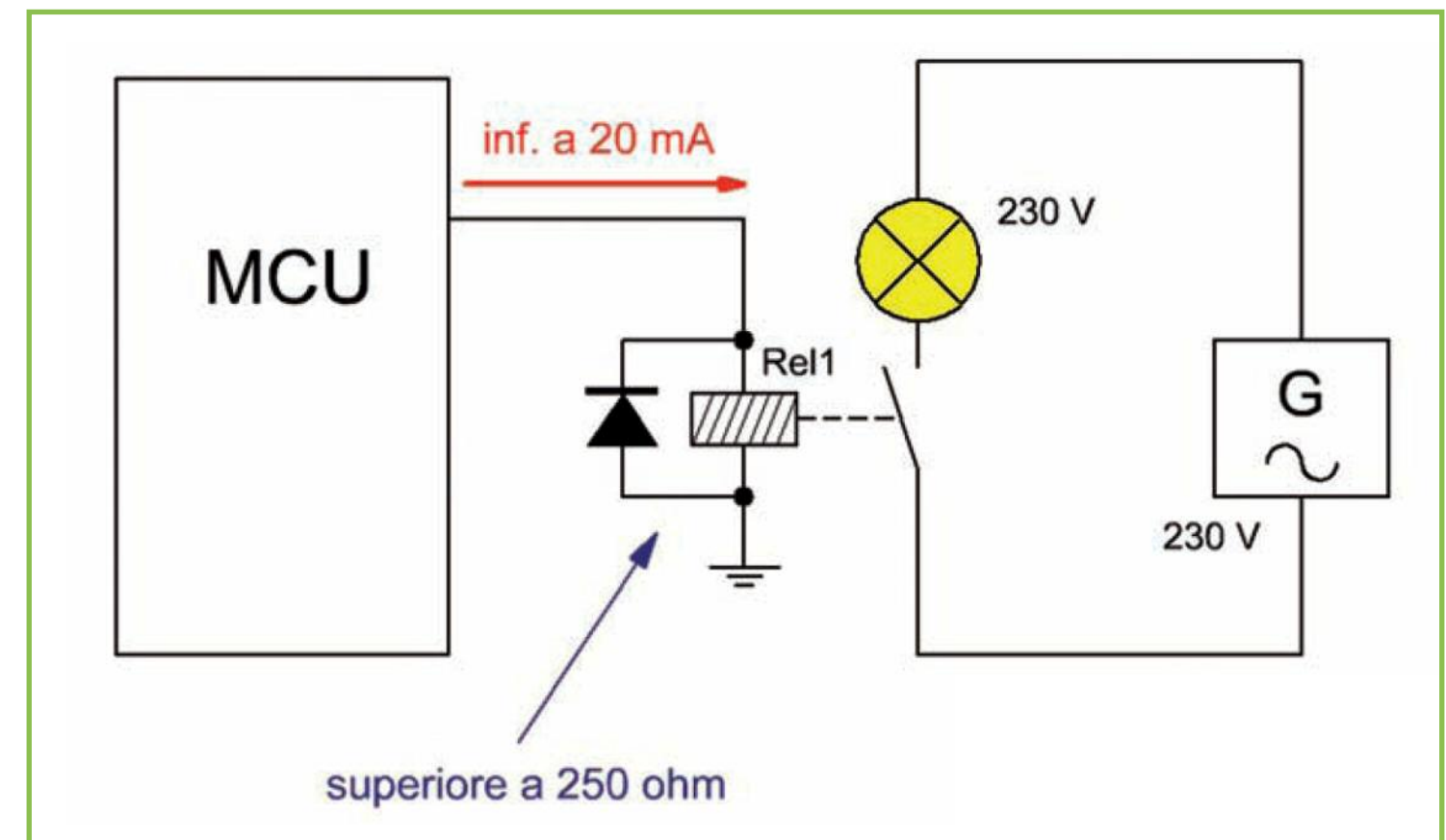



Figura 6: un microcontrollore pilota una lampada in AC a 230 V utilizzando un Relè (senza transistor).

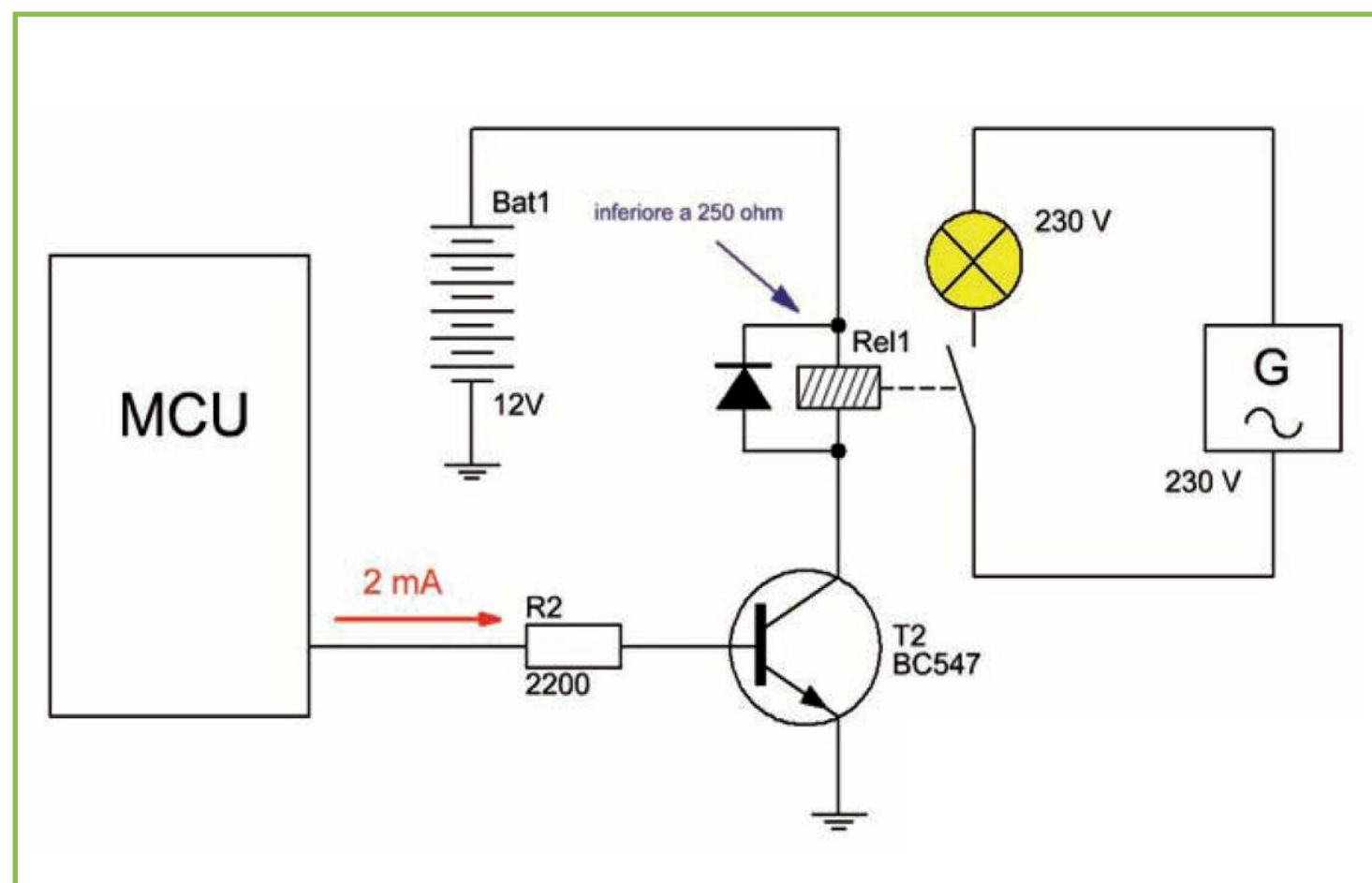



Figura 7: un microcontrollore pilota una lampada in AC a 230 V utilizzando un Relè (con transistor).

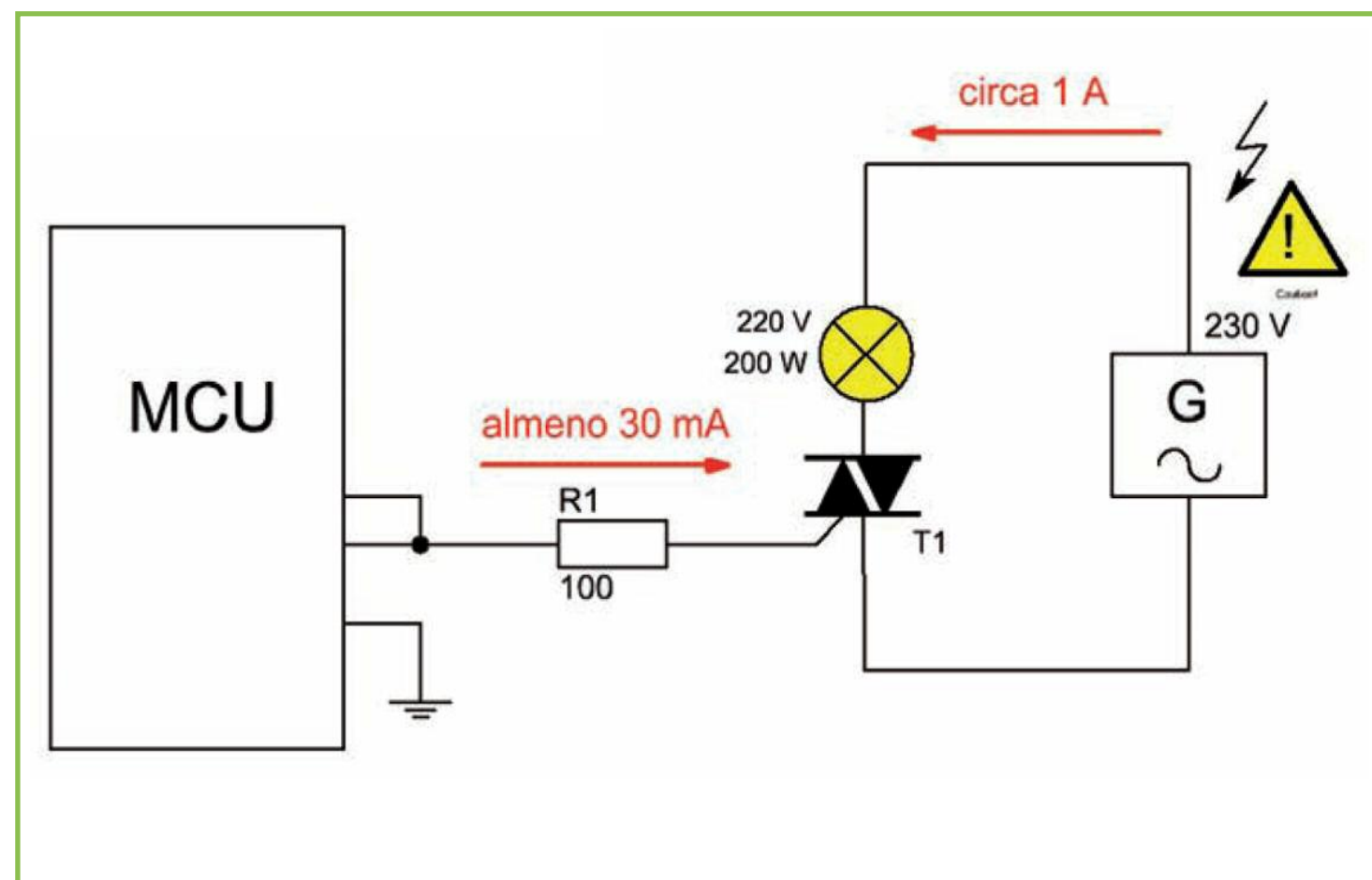



Figura 8: un microcontrollore pilota una lampada in AC a 230 V utilizzando un Triac.

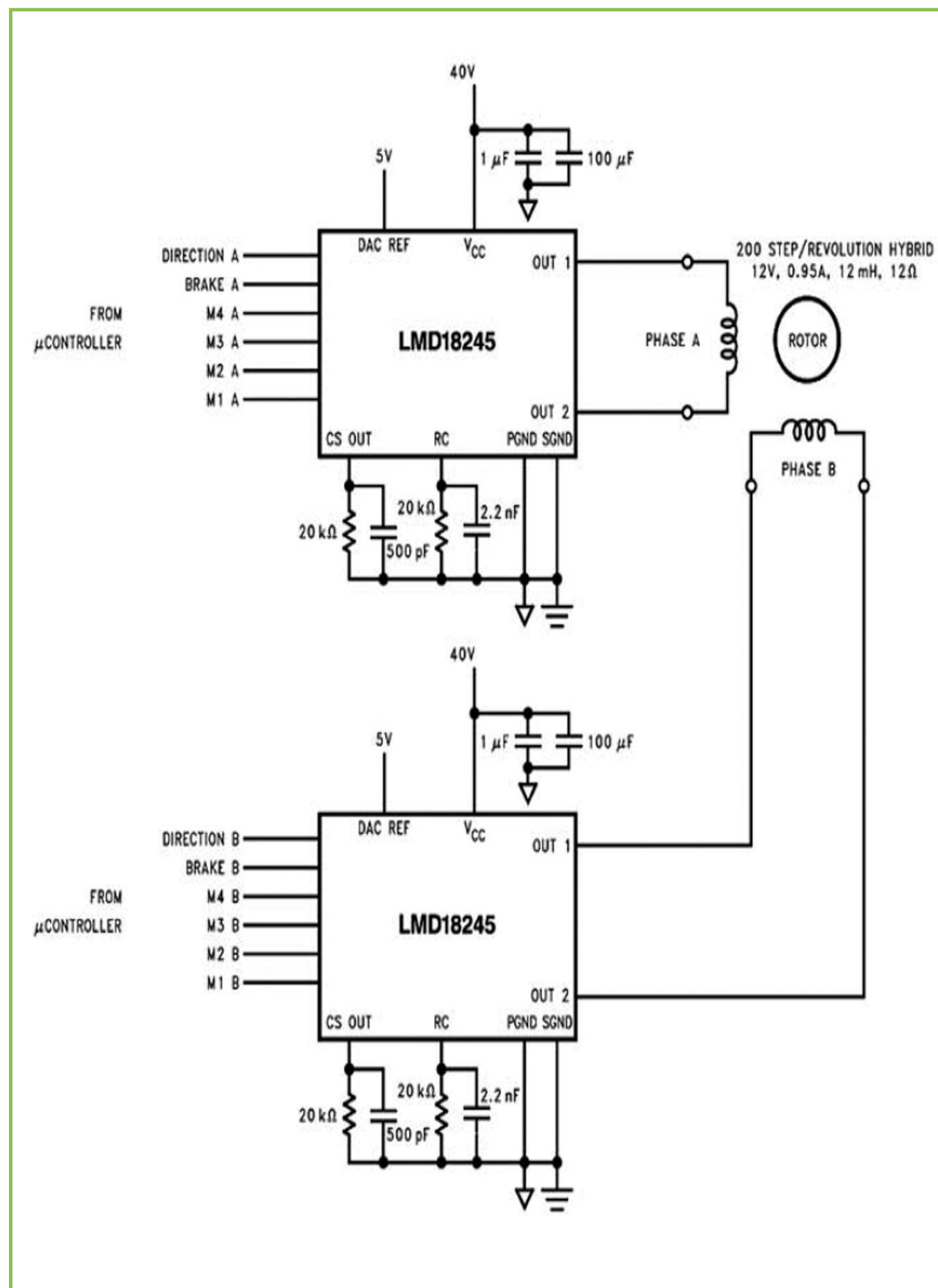



Figura 9: un microcontrollore pilota un motore utilizzando l'integrato LMD18245.

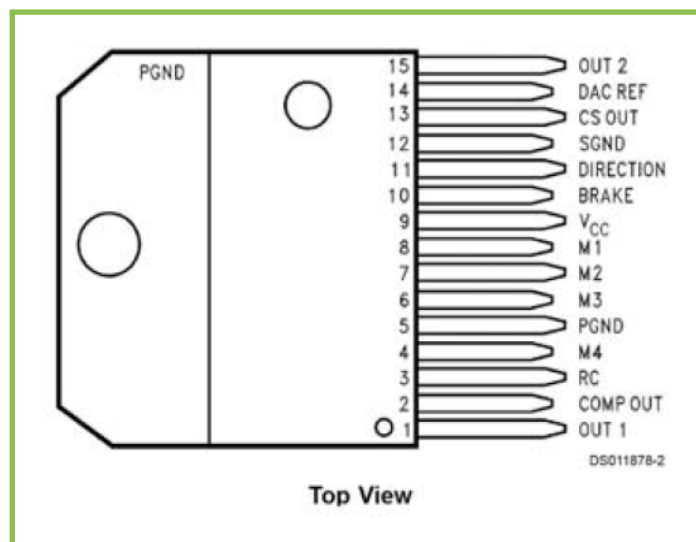



Figura 10: vista superiore dell'integrato LMD18245

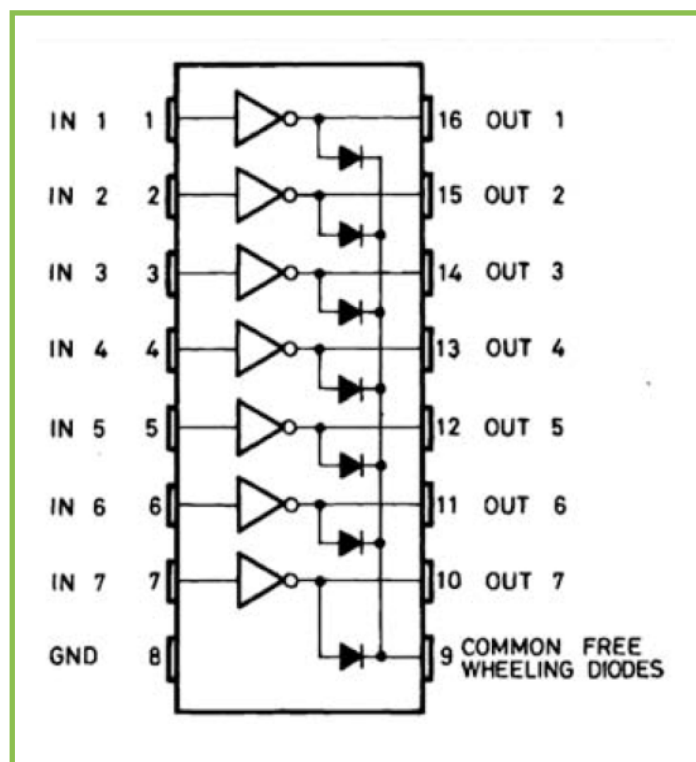



Figura 11: connessione dei pin dell'integrato ULN2001 (top view).

UTILIZZO DEL MOSFET IN PARALLELO

Quando la corrente si fa veramente alta, si possono usare tanti Mosfet collegati in parallelo. Si moltiplica in questa maniera la correte sopportata da ogni dispositivo, consentendo il passaggio di correnti molto forti. Il piedino di uscita del microcontrollore, anche in questo caso, non viene

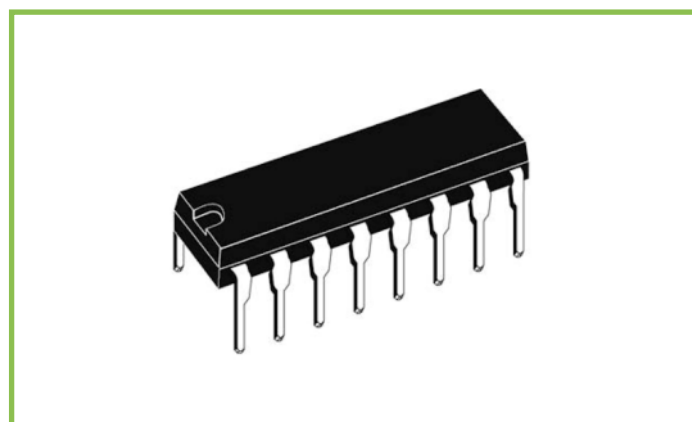



Figura 12: contenitore DIP 16 dell'integrato ULN2001.

coinvolto nel passaggio di corrente, e la sua azione avviene solo in tensione. L'esempio che segue prevede un microcontrollore capace di pilotare un carico resistivo che assorbe ben 60 A. Una corrente da far paura.
Un utilizzo tipico di questo metodo è quello del motore a chopper, nel quale la corrente allo spunto può superare facilmente i 100 A. Naturalmente lo schema proposto è teorico e di uso generico, ma comunque funzionante.

UTILIZZO DEL RELÈ

Anche il relè può essere utilizzato proficuamente con un microcontrollore. In questa maniera è anche possibile comandare i carichi a 230 V dal momento che la sua bobina e l'interruttore a scatto sono isolati tra loro. Si possono avere due casi possibili:

- il relè ha una bobina con valore ohmico superiore a 250 ohm e la sua tensione di attivazione è di 5 V. In questo caso è possibile alimentarlo direttamente attraverso la porta di uscita del microcontrollore.
- Il relè ha una bobina con valore ohmico inferiore a 250 ohm e funziona con una tensione maggiore di 5 V. Il suo assorbimento supera pertanto i 20 mA. In questo caso è necessario l'utilizzo di un transistor. In ogni caso è necessario prevedere l'uso di un diodo collegato in anti-parallelo tra i contatti della bobina per prevenire le extratensioni, pericolose per i componenti adiacenti. Per la sua natura meccanica il relé ha un tempo di commutazione estremamente lento.

UTILIZZO DEL TRIAC

Con l'utilizzo del Triac è possibile comandare i carichi a 230 V. In questa maniera, lampade e carichi resistivi possono essere pilotati attraverso un microcontrollore, senza usare componenti meccanici ma elettronici. Anche qui è necessario rispettare alcune regole:

- la velocità di commutazione del Triac potrebbe essere alquanto bassa;
- il Triac deve sopportare la corrente e la tensione richiesta dal carico;
- la corrente di Gate potrebbe essere elevata, anche superiore a 50 ÷ 100 mA. In questo caso si devono utilizzare contemporaneamente più porte del micro oppure dei buffers per amplificare la corrente d'uscita;
- usando la tensione di rete a 230 V sarebbe opportuno opto-isolare il circuito. L'esempio prevede una lampada a bassa potenza di 200 W ma alimentata a 230 V. Se la corrente di Gate è sufficiente ad innescare il Triac, la conduzione dello stesso è assicurata.

UTILIZZO DELL'INTEGRATO DI POTENZA LMD18245

L'integrato LMD18245 è un amplificatore di potenza a ponte che controlla in corrente i

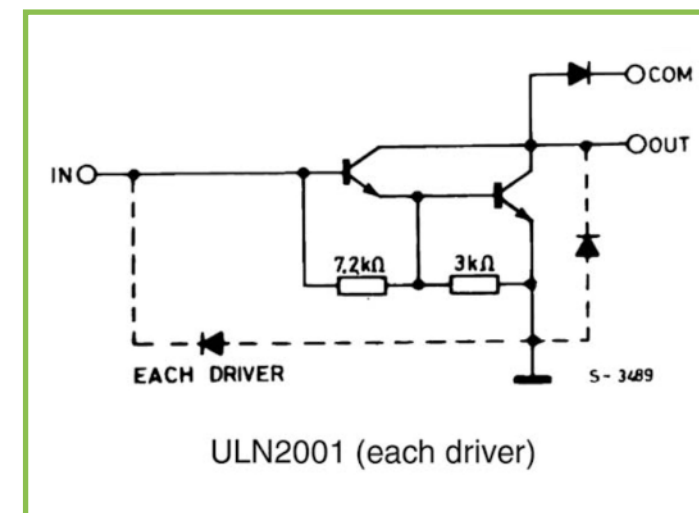



Figura 13: configurazione di un elemento della rete ULN2001.

motori DC a spazzole e i motori passo passo a singola fase. Per questo tipo di applicazione ovviamente è richiesta la creazione di un software più complesso per il microcontrollore, al fine di pilotare correttamente le porte adibite alle operazioni.

UTILIZZO DELL'INTEGRATO ULN2001 COME RETE DI DARLINGTON

Con questo metodo viene ridotto drasticamente lo spazio occupato dai componenti elettronici e di conseguenza l'intero prototipo assume dimensioni molto compatte. Si tratta di un array di Darlington funzionante ad alta tensione ed elevata corrente a collettore aperto, contenuti in un unico integrato composto da sette elementi. Ogni elemento è capace di erogare la corrente di 500 mA (con picchi di 600 mA). In aggiunta, anche i 7 diodi di protezione da extratensione sono inclusi nel dispositivo. Questo integrato prevede una disposizione dei suoi pin molto intelligente, infatti le sette uscite sono collocate in posizione opposta ai pin d'ingresso, per permetterne una semplice implementazione su PCB.

IEshop

ANNATA 2012 DI FIRMWARE

Un file zip che contiene tutte le riviste del 2012 in formato pdf ad alta risoluzione, tutti i downloads realtivi ai singoli numeri. E' incluso anche Acrobat Reader 6 per sfogliare le riviste.

Prezzo: € 18.08

READY FOR PIC

NOVITÀ

Scheda di sviluppo applicazioni con microcontrollori che utilizzano MCU PIC a 28/40 pin - con PIC18F45K22 a 40 pin. La miglior soluzione per lo sviluppo rapido e semplice di applicazioni con microcontrollori che utilizzano PIC MCU a 28 e 40 pin. La scheda viene fornita con PIC18F45K22 collegato al cristallo oscillatore a 8MHz. Contiene quattro header IDC10 per tutte le porte dei microcontrollori disponibili, modulo USB-UART, area di prototipazione e di un circuito di alimentazione. La scheda è appositamente progettata per adattarsi in un particolare involucro di plastica bianca che viene venduta separatamente. La scheda viene preprogrammata con un veloce bootloader UART che elimina la necessità di programmatori esterni. La scheda supporta anche il veloce programmatore USB 2.0 mikroProg con hardware In-Circuit Debugger.

Prezzo: € 28.06

OFFERTA

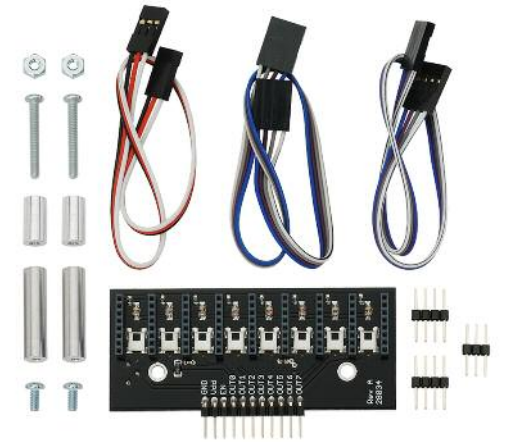
MODULO DI CONVERSIONE RS232/BLUETOOTH IN CLASSE 2

F2M03C2 è un modulo Bluetooth SMD operante in classe 2 adatto per comunicazioni dati e voce, si interfaccia via UART o USB e consente la trasmissione dati fino al massimo datarate consentito da Bluetooth (723.2Kbps).
Il dispositivo è dotato di una interfaccia audio digitale 13-bit PCM bidirezionale (campionamento a 8kHz) oltre a vari I/O analogici e digitali ed un'interfaccia I2C.

Prezzo: € ~~67.10~~ € 41.48

KIT LINE FOLLOWING A INFRAROSSI

L'Infrared Line Follower Kit fornisce otto emettitori a infrarossi e una coppia di ricevitori per le applicazioni di "Line Following" ad alta precisione. E' incluso l'hardware di montaggio sia per il Boe-Bot che per Stingray, per una facile messa in servizio.

Prezzo: € 39.04

BUSSOLA ELETTRONICA

NOVITÀ

Compass Click è una scheda accessoria che utilizza il nuovo socket mikroBUS. È dotata di una bussola elettronica LSM303DLHC ultra compatto ad alte prestazioni. Esso comprende uno specifico elemento di rilevamento e un'interfaccia IC in grado di misurare sia l'accelerazione lineare (in scala di ± 2g / ± 4g / ± 8g / ± 16g) e il campo magnetico (fondo scala da ± 1,3 a ± 8,1 gauss) e fornisce un uscita dati a 16-bit tramite l'interfaccia I2C. Le letture ricevute dalla IC possono essere elaborati ed utilizzati per determinare il polo nord terrestre. La scheda è stato progettata per usare solo l'alimentazione 3.3V. Tutte queste caratteristiche rendono questa scheda ideale per la compensazione della bussola, rilevamento di posizione, rotazione della mappa, pedometro, display di orientamento e molto altro.

Prezzo: € 23.18

ARDUINO TFT LCD

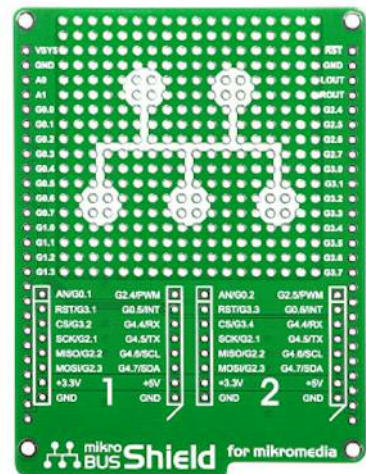
PRIMO PIANO

Il display Arduino Graphic LCD (GLCD) è uno schermo TFT LCD retroilluminato con headers. E' possibile rappresentare sullo schermo testi immagini e forme tramite le librerie GLCD. Sul retro del display è montato un lettore micro-SD dove è possibile, tra le altre cose, salvare immagini bitmap riproducibili sullo schermo. Gli headers sul modulo sono pensati per essere inseriti nel socket sul fronte della Arduino Esplora ma sono anche compatibili con qualunque Arduino AVR-based (Uno, Leonardo, etc.).

Prezzo: € 23.18

DISPLAY OLED 96X64 INTELLIGENTE

Il microOLED μOLED-96 è un display grafico a colori completo di scheda di controllo che può essere gestito direttamente da un microcontrollore (PIC, AVR, ARM, Stamp) via porta seriale. Possibilità di scegliere 3 tipi di dimensioni per il font (5x7, 8x8 and 8x12) e con appositi comandi disegnare cerchi, linee, rettangoli, scegliere il colore dello sfondo e caricare immagini bitmap. A bordo adattore memory card per micro-SD (μSD) per immagazzinare icone, immagini, animazioni ecc..

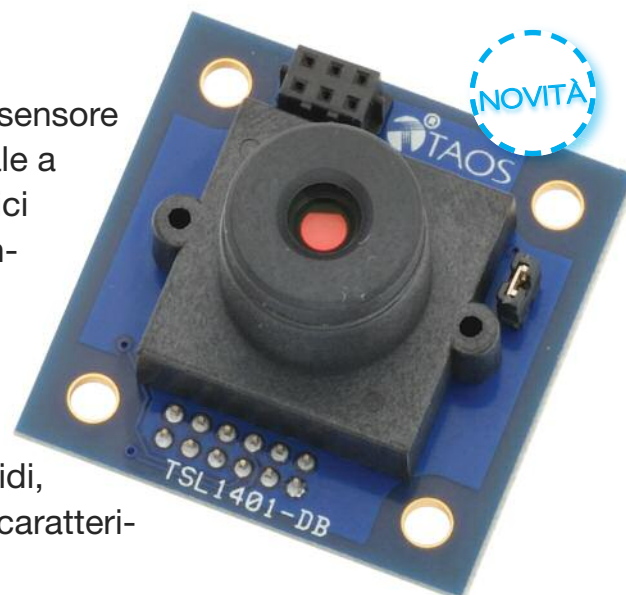
Prezzo: € ~~47.58~~ € 35.38

MIKROBUS SHIELD FOR MIKROMEDIA

La mikroBUS Shield è una scheda di espansione pin-compatibile con tutte le schede mikromedia, che dispone di due connettori host mikroBUS che consentono di connettere decine di schede Click. Grazie a questa scheda, è possibile fornire ulteriori funzionalità alla vostra scheda mikromedia. Una zona Breadboard di prototipazione è disponibile anche per l'inserimento componenti aggiuntivi ed espandere le funzionalità di base con caratteristiche personalizzate. La scheda viene fornita con due socket 1x26 pin femmina, e anche quattro connettori femmina 1x8 per connettori host mikroBUS.

Prezzo: € 6.83

SCHEDA PER SCANSIONE IMMAGINI

Il TSL1401Linescan Imaging Daughterboard (scheda sensore per scansioni immagini) offre una vista unidimensionale a quasi qualsiasi microcontrollore. Produce dati analogici cadenzati, i cui livelli di tensione corrispondono alla intensità della luce in ogni pixel.

Mediante un convertitore analogico-digitale (o anche una semplice soglia logica digitale), i dati dell'immagine sono facilmente trasferiti ad un microcontrollore per rilevare oggetti, spigoli, divergenze, fori, livelli liquidi, texture, fonti emissive semplici, codici a barre e altre caratteristiche visibili.

Prezzo: € 47.58

CONNECTEVE - GRAFICA INCREDIBILE ANCHE SU MCU A 8 BIT

La scheda ConnectEVE fornisce grafica veloce e un forte impatto visivo, anche con MCU a 8-bit con poca memoria Flash e RAM. Grazie al controller grafico FT800 (che è attualmente lo stato dell'arte) integra display, audio e touch a basso costo in una soluzione single-chip, facile da usare. La FT800/EVE ha una interfaccia SPI con il supporto di interrupt e l'uscita audio mono. E' fornita di header IDC10 e 1x10 che possono essere utilizzati per collegare le vostre schede di sviluppo o progetti finali esistenti.

Prezzo: € 67.10

LCD TFT 4,3" TOUCH PER BEAGLE BONE BLACK

Il 4DCAPE-43T è specificamente progettata per la Beagle Bone Black (BBB), e fornisce un display da 4,3" per la BBB per l'interazione diretta dell'utente e la visualizzazione delle informazioni. Disponibile sia in touch resistivo (4DCAPE-43T) e non-touch (4DCAPE-43), il CAPE dispone di un display TFT LCD da 4,3" 480x272 di risoluzione e sette pulsanti. La Beagle Bone Black si collega direttamente alla parte posteriore del 4DCAPE-43, e fornisce tutti i segnali richiesti, come il segnale del display e l'alimentazione. La scheda comprende anche quattro fori di fissaggio per un facile montaggio. Si prega di notare che la 4DCAPE-43 è stato progettata per essere il display primario solo per Beagle Beagle Bone Black e non può essere essere programmata utilizzando sistemi 4D Systems Workshop 4 IDE.

Prezzo: € 59.78

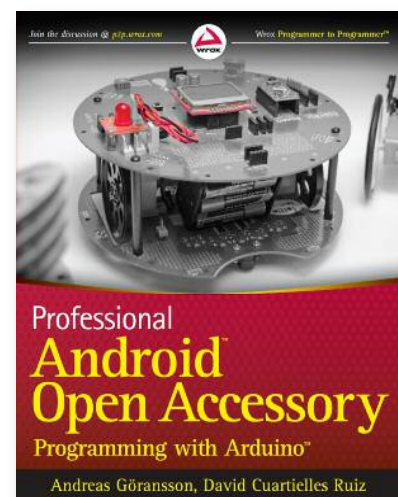
OSCILLOSCOPIO DIGITALE

Il ScopeLogic DAQ è un sistema completo di acquisizione dati, composto da un oscilloscopio digitale e un analizzatore logico. Questo piccolo strumento permette un'accurata analisi dei segnali analogici e digitali, grazie ad un campionamento simultaneo, rendendo più semplice la relazione tra i segnali rilevati. E 'possibile utilizzare contemporaneamente 16 canali digitali e 2 canali analogici standard. La frequenza massima di campionamento durante l'utilizzo di tutti i canali è di 100MHz, è possibile aumentare le frequenze di campionamento fino a 200MHz riducendo il numero di canali utilizzati. Inoltre l'analizzatore logico offre la possibilità di registrare i segnali con la

regolazione del tempo, cambiare il livello di soglia. La versione base dello ScopeLogic DAQ è fornita con una memoria dati di 128K, che può essere aumentata fini a 512Kb (opzionale).

 Prezzo: € ~~890.60~~ € 608.78

LIBRO ANDROID OPEN ACCESSORY

Impara come controllare la tua casa o l'auto dal tuo smartphone Android: aria condizionata, luci, sistemi di intrattenimento, e molto altro! Android Open Accessory è un nuovo, semplice e sicuro protocollo per il collegamento di qualsiasi dispositivo a microcontrollore con un smartphone Android o tablet. Questa guida, in inglese, mostra ai programmatori di Android come utilizzare Android Open Accessory con Arduino, per controllare sistemi come illuminazione, aria condizionata e sistemi di intrattenimento da dispositivi Android. Inoltre, insegna le tecniche necessarie per creare giochi e prodotti pratici che sfruttano la tecnologia Android. Inoltre, il volume, introduce all'Android Open Accessory e mostra come configurare l'hardware e l'ambiente di sviluppo

Viene illustrato come codificare elementi Android e Arduino di un elemento. Vengono illustrate le caratteristiche di quattro progetti completi dove è possibile utilizzare vari sensori e indicatori/attuatori, è incluso anche il codice sorgente. Infine, fornisce agli sviluppatori Android gli strumenti per creare potenti, sofisticati progetti. Questo libro aprirà sicuramente nuove opportunità per gli sviluppatori di Android. 408 pagine.

 Prezzo: € 36.00

- Sei già cliente? Risparmierai il 10% sul tuo nuovo ordine!

- Il tuo ordine supera i 200 EUR? Le spese di trasporto sono Gratis!

- Vuoi ricevere un buono sconto di 5 EUR? Recensisci i prodotti acquistati!

- Vuoi ricevere particolari offerte o promozioni? Diventa membro Inware Edizioni.

Fai il tuo ordine oggi stesso su